# The categorical contours of the Chomsky-Schützenberger representation theorem

Paul-André Melliès, Noam Zeilberger

# THE CATEGORICAL CONTOURS OF THE CHOMSKY-SCHÜTZENBERGER REPRESENTATION THEOREM

PAUL-ANDRÉ MELLIÈS AND NOAM ZEILBERGER

[a] IRIF, Université Paris Cité, CNRS, Inria, Paris, France
*e-mail address*: paul-andre.mellies@cnrs.fr

[b] LIX, École Polytechnique, Inria, Palaiseau, France
*e-mail address*: noam.zeilberger@polytechnique.edu

Abstract. We develop fibrational perspectives on context-free grammars and on finite state automata over categories and operads. A generalized CFG is a functor from a free colored operad (= multicategory) generated by a pointed finite species into an arbitrary base operad: this encompasses classical CFGs by taking the base to be a certain operad constructed from a free monoid, as an instance of a more general construction of an *operad of spliced arrows* $\mathcal{W}\mathcal{C}$ for any category $\mathcal{C}$. A generalized NDFA is a functor satisfying the unique lifting of factorizations and finite fiber properties, from an arbitrary bipointed category or pointed operad: this encompasses classical word automata and tree automata without $\epsilon$-transitions, but also automata over non-free categories and operads. We show that generalized context-free and regular languages satisfy suitable generalizations of many of the usual closure properties, and in particular we give a simple conceptual proof that context-free languages are closed under intersection with regular languages. Finally, we observe that the splicing functor $\mathcal{W} : \mathrm{Cat} \to \mathrm{Oper}$ admits a left adjoint $\mathcal{C} : \mathrm{Oper} \to \mathrm{Cat}$, which we call the *contour category* construction since the arrows of $\mathcal{C}\,\mathcal{O}$ have a geometric interpretation as oriented contours of operations of $\mathcal{O}$. A direct consequence of the contour / splicing adjunction is that every pointed finite species induces a universal CFG generating a language of *tree contour words*. This leads us to a generalization of the Chomsky-Schützenberger Representation Theorem, establishing that a subset of a homset $L \subseteq \mathcal{C}(A, B)$ is a CFL of arrows iff it is a functorial image of the intersection of a $\mathcal{C}$-chromatic tree contour language with a regular language.

## Introduction

In "Functors are Type Refinement Systems" [MZ15], we argued for the idea that rather than being modelled merely as categories, type systems should be modelled as functors $p : \mathcal{D} \to \mathcal{T}$ from a category $\mathcal{D}$ whose morphisms may be considered as abstract typing derivations to

*Key words and phrases:* category theory, operads, context-free grammars and context-free languages, finite-state automata and regular languages, tree contour words, representation theorem.

This is a thoroughly revised and expanded version of a paper with a similar title [MZ22] presented at the 38th Conference on the Mathematical Foundations of Programming Semantics (MFPS 2022).

The first author is partially supported by ANR ReciProg (ANR-21-CE48-0019). The second author is partially supported by ANR LambdaComb (ANR-21-CE48-0017).

a category $\mathcal{T}$ whose morphisms are the terms corresponding to the underlying *subjects* of those derivations. One consequence of this fibrational point of view is that the notion of typing judgment receives a simple mathematical status, as a triple $(R, f, S)$ consisting of two objects $R, S$ in $\mathcal{D}$ and a morphism $f$ in $\mathcal{T}$ such that $p(R) = \mathrm{dom}(f)$ and $p(S) = \mathrm{cod}(f)$. The question of finding a derivation for such a judgment then reduces to the "lifting problem" of finding a morphism $\alpha : R \to S$ such that $p(\alpha) = f$.

We developed this perspective in a series of papers [MZ13, MZ15, MZ18, MZ16], and believe that it may be usefully applied to a large variety of deductive systems, beyond type systems in the traditional sense. In this work, we focus on derivability in context-free grammars and on recognition by non-deterministic finite-state automata – two classic topics in formal language theory with wide applications in computer science. Although grammars and automata are of a very different nature, both may be naturally described as certain kinds of functors.

Free operads[1] play an important role in our treatment of context-free grammars. Indeed, we will ultimately propose that a "generalized CFG" may be defined simply as *a functor from a free operad generated by a pointed finite species* into an arbitrary base operad. In order to recover the classical definition of CFG, we rely on a construction $\mathcal{W} : \mathrm{Cat} \to \mathrm{Oper}$ turning any category $\mathcal{C}$ into an operad $\mathcal{W}\mathcal{C}$, which we call its *operad of spliced arrows*. One recovers the classical definition by taking $\mathcal{C} = \mathcal{F}\ \mathbb{B}_\Sigma$ to be the one-object category freely generated from the one-node "bouquet" graph with a loop $a : * \to *$ for every letter of the alphabet $a \in \Sigma$. However, as we will see, it is very natural to consider context-free languages of arrows in an arbitrary category, or even generalized context-free languages of constants in an arbitrary operad.

Non-deterministic finite-state automata, it turns out, may be naturally modelled as *functors satisfying the unique lifting of factorizations and finite fiber properties*. One nice aspect of this formulation is that it captures both word automata and tree automata, which correspond to functors satisfying these two properties into free categories and into free operads respectively. However, again we will see that it is very natural and at times essential to work with a more general definition of NDFA over a non-free category or operad.

A benefit of expressing both context-free grammars and non-deterministic finite state automata as functors is that they are placed within a common framework, which facilitates comparison and combination. The germ of this paper was planted early on in the wake of our type refinement systems work, in 2017, when we began considering context-free grammars as functors of multicategories. But ultimately it was only in 2022 that the key mathematical definitions emerged very quickly (over the span of a couple weeks), when we mused that it could be interesting to analyze the Chomsky-Schützenberger Representation Theorem, a classic result in formal language theory that in its now-standard formulation states that a language is context-free if and only if it is a homomorphic image of the intersection of a Dyck language of well-bracketed words with a regular language. One reason the theorem is interesting is that it invokes a non-trivial closure property of context-free languages, namely closure under intersection with regular languages. Another reason is that it suggests, intuitively, that Dyck languages are in some sense "universal" context-free languages.

We will see that both of these aspects of the theorem have good categorical explanations. First, the intersection of a context-free language with a regular language may be computed by

[1] In this paper, when we say "operad" we always mean *colored operad*, in the sense that our operads always carry a (potentially trivial) set of colors and operations are sorted $f : A_1, \ldots, A_n \to B$ in addition to having an arity $n$. An operad in this sense is also called a *multicategory*, and we use the two words interchangeably.

first taking the *pullback* of the functor representing the corresponding context-free grammar along the corresponding non-deterministic finite state automaton, which for non-trivial but elementary reasons defines a context-free grammar over the *runs* of the automaton. Second, the aforementioned "spliced arrows" construction $\mathcal{W}$ extends to an *adjunction* between the category of categories and the category of operads,

$$\mathrm{Oper} \underset{\mathcal{W}}{\overset{\mathcal{C}}{\underset{\longleftarrow}{\overset{\longrightarrow}{\bot}}}} \mathrm{Cat}$$

where the left adjoint builds what we call the "contour category" $\mathcal{C}\,\mathcal{O}$ of an operad $\mathcal{O}$. This contour / splicing adjunction, which seems to be of independent interest, has as an immediate consequence that every pointed finite species induces a universal CFG generating a language of *tree contour words.* Such contour words are closely related to Dyck words, and allow us to prove a generalization of the representation theorem for context-free languages of arrows in an arbitrary category.

The rest of the paper is organized as follows. Section 1 is devoted to context-free grammars, Section 2 to non-deterministic finite state automata, and Section 3 to the representation theorem. We discuss related work at the end of each of these sections, and we conclude in Section 4. Finally, an Addendum (not in the original conference version of the paper) begins to develop a more abstract perspective on generalized context-free languages as *initial models* of generalized context-free grammars, providing minimal solutions to systems of polynomial equations with a fibrational interpretation.

## 1. Context-free languages of arrows in a category

In this section, after some preliminaries on free operads and species, we describe the operad of spliced arrows construction, explain how to use it to define context-free grammars over arbitrary categories generating *context-free languages of arrows,* and show that many standard properties of CFGs and CFLs may be naturally reformulated in this general setting. We also spend some time developing the fibrational view of parsing as a lifting problem, explaining how any functor from a free operad $\mathcal{F}\,\mathbb{S} \to \mathcal{O}$ may be represented as a "displayed free operad" corresponding to a lax functor $\mathcal{O} \to \mathrm{Span}(\mathrm{Set})$ that admits a certain inductive definition. Finally, we describe how the definition of categorical CFG admits a further natural generalization by allowing the base operad to be arbitrary, and show that this encompasses several extensions of the notion of context-free grammar from the literature.

### 1.1. **Preliminaries on free operads and species.**

We assume some basic familiarity with operads, otherwise known as multicategories, as introduced for example in Chapter 2 of Leinster's book [Lei04]. We write $f \circ (g_1, \ldots, g_n)$ for parallel composition of operations, and $f \circ_i g$ for partial composition *after* the first $i$ inputs (so $f \circ_0 g$ denotes composition of $g$ into the 0th input of $f$). Here we recall the notion of a (colored non-symmetric) species, and how one gives rise to a free operad.

A *colored non-symmetric species,* which we abbreviate to **species**[2] for short, is a tuple $\mathbb{S} = (C, V, i, o)$ consisting of a span of sets $C^* \xleftarrow{i} V \xrightarrow{o} C$ with the following interpretation: $C$ is a set of "colors", $V$ is a set of "nodes", and the functions $i : V \to C^*$ and $o : V \to C$ return respectively the list of input colors and the unique output color of each node. Adopting the same notation as we use for operations of an operad, we write $x : R_1, \ldots, R_n \to R$ to indicate that $x \in V$ is a node with list of input colors $i(x) = (R_1, \ldots, R_n)$ and output color $o(x) = R$. However, it should be emphasized that a species by itself only contains bare coloring information about the nodes, and does not say how to compose them as operations.

We will be primarily interested in *finite* species. We say that a species is finite (also called *polynomial* [Joy86]) just in case both sets $C$ and $V$ are finite.

A map of species $\phi : \mathbb{S} \to \mathbb{R}$ from $\mathbb{S} = (C, V, i, o)$ to $\mathbb{R} = (D, W, i', o')$ is given by a pair $\phi = (\phi_C, \phi_V)$ of functions $\phi_C : C \to D$ and $\phi_V : V \to W$ making the diagram commute:

$$\begin{array}{ccccc} C^* & \xleftarrow{i} & V & \xrightarrow{o} & C \\ \downarrow{\scriptstyle \phi_C^*} & & \downarrow{\scriptstyle \phi_V} & & \downarrow{\scriptstyle \phi_C} \\ D^* & \xleftarrow{i'} & W & \xrightarrow{o'} & D \end{array}$$

Equivalently, overloading $\phi$ for both $\phi_C$ and $\phi_V$, every node $x : R_1, \ldots, R_n \to R$ of $\mathbb{S}$ must be sent to a node $\phi(x) : \phi(R_1), \ldots, \phi(R_n) \to \phi(R)$ of $\mathbb{R}$. Every operad $\mathcal{O}$ has an underlying species with the same colors and whose nodes are the operations of $\mathcal{O}$, and this extends to a forgetful functor $\mathbb{U} : \mathrm{Oper} \to \mathrm{Spec}$ from the category of operads and functors of operads to the category of species and maps of species. Moreover, this forgetful functor has a left adjoint

$$\mathrm{Spec} \underset{\mathbb{U}}{\overset{\mathcal{F}}{\underset{\longleftarrow}{\overset{\longrightarrow}{\bot}}}} \mathrm{Oper}$$

which sends any species $\mathbb{S}$ to a free operad with the same set of colors and whose operations are generated from the nodes of $\mathbb{S}$. By the universal property of the adjoint pair, there is a natural isomorphism of homsets

$$\mathrm{Oper}(\mathcal{F}\ \mathbb{S}, \mathcal{O}) \cong \mathrm{Spec}(\mathbb{S}, \mathbb{U}\,\mathcal{O})$$

placing functors of operads $p : \mathcal{F}\ \mathbb{S} \to \mathcal{O}$ and maps of species $\phi : \mathbb{S} \to \mathbb{U}\,\mathcal{O}$ in one-to-one correspondence. In the sequel, we will leave the action of the forgetful functor implicit, writing $\mathcal{O}$ for both an operad and its underlying species $\mathbb{U}\,\mathcal{O}$.

### 1.2. **The operad of spliced arrows of a category.**

**Definition 1.1.** Let $\mathcal{C}$ be a category. The **operad $\mathcal{W}\mathcal{C}$ of spliced arrows in $\mathcal{C}$** is defined as follows:

- its colors are pairs $(A, B)$ of objects of $\mathcal{C}$;
- its $n$-ary operations $(A_1, B_1), \ldots, (A_n, B_n) \to (A, B)$ consist of sequences $w_0 - w_1 - \ldots - w_n$ of $n + 1$ arrows in $\mathcal{C}$ separated by $n$ gaps notated $-$, where each arrow must have type $w_i : B_i \to A_{i+1}$ for $0 \leq i \leq n$, under the convention that $B_0 = A$ and $A_{n+1} = B$;

[2]Species in this sense could also be called signatures, and are sometimes called "multigraphs" [Lam89, Wal89] since they bear the same relationship to multicategories as graphs do to categories (though that terminology has an unfortunate clash with a different concept in graph theory). We use "species" to emphasize the link with Joyal's theory of (symmetric) species [Joy81] and with generalized species [FGHW08].

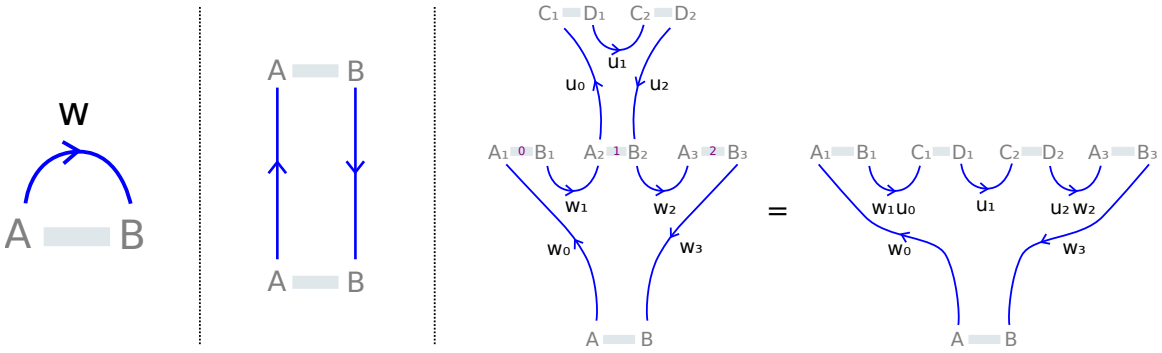



Figure 1: Left: a constant of $\mathcal{W}\mathcal{C}$. Middle: an identity operation. Right: example of partial composition. Here we compose $g = u_0{-}u_1{-}u_2 : (C_1, D_1), (C_2, D_2) \to (A_2, B_2)$ into $f = w_0{-}w_1{-}w_2{-}w_3 : (A_1, B_1), (A_2, B_2), (A_3, B_3) \to (A, B)$ at the gap labelled 1 to obtain $f \circ_1 g = w_0{-}w_1u_0{-}u_1{-}u_2w_2{-}w_3 : (A_1, B_1), (C_1, D_1), (C_2, D_2), (A_3, B_3) \to (A, B)$.

- composition of spliced arrows is performed by "splicing into the gaps": formally, the partial composition $f \circ_i g$ of a spliced arrow $g = u_0{-}\ldots{-}u_m$ into another spliced arrow $f = w_0{-}\ldots{-}w_n$ is defined by substituting $g$ for the $i$th occurrence of $-$ in $f$ (starting from the left using 0-indexing) and interpreting juxtaposition by sequential composition in $\mathcal{C}$ (see Fig. 1 for an illustration);
- the identity operation on $(A, B)$ is given by $id_A{-}id_B$.

It is routine to check that $\mathcal{W}\mathcal{C}$ satisfies the associativity and neutrality axioms of an operad, these reducing to associativity and neutrality of composition of arrows in $\mathcal{C}$. Indeed, the spliced arrows operad construction defines a functor $\mathcal{W}- : \mathrm{Cat} \to \mathrm{Oper}$ since any functor of categories $F : \mathcal{C} \to \mathcal{D}$ induces a functor of operads $\mathcal{W}F : \mathcal{W}\mathcal{C} \to \mathcal{W}\mathcal{D}$, acting on colors by $(A, B) \mapsto (FA, FB)$ and on operations by $w_0{-}\ldots{-}w_n \mapsto Fw_0{-}\ldots{-}Fw_n$.

**Example 1.2.** As referenced in the Introduction, for any alphabet $\Sigma$ there is a one-object category $\mathcal{F}\,\mathbb{B}_\Sigma$, freely generated from the bouquet graph with a loop $a : * \to *$ for every letter of $a \in \Sigma$, so that words $w \in \Sigma^*$ may be regarded as arrows $w : * \to *$ in $\mathcal{F}\,\mathbb{B}_\Sigma$. In this case, the construction yields an operad $\mathcal{W}\Sigma := \mathcal{W}\mathcal{F}\,\mathbb{B}_\Sigma$ that we like to call the **operad of spliced words** in $\Sigma$. It has a single color, and its $n$-ary operations are sequences $w_0{-}\ldots{-}w_n$ of $n+1$ words over $\Sigma$. For example, in the operad of spliced words over the ASCII alphabet, we have that `"Hell"−", "−"rld!"` $\circ$ `("o","Wo")` $=$ `"Hello, World!"`.

**Remark 1.3.** Although $\mathcal{F}\,\mathbb{B}_\Sigma$ is a free category, this property of being freely generated does not extend to its operad of spliced words. Indeed, an operad of spliced arrows $\mathcal{W}\mathcal{C}$ is almost *never* a free operad. That's because any pair of objects $A$ and $B$ induces a binary operation $id_A{-}id_A{-}id_B : (A, A), (A, B) \to (A, B)$, and any arrow $w : A \to B$ of $\mathcal{C}$ induces a corresponding constant $w : (A, B)$. Since $id_A{-}id_A{-}id_B \circ (id_A, w) = w$, $\mathcal{W}\mathcal{C}$ cannot be a free operad except in the trivial case where $\mathcal{C}$ has no objects and no arrows.

**Example 1.4.** Arrows in free categories with more than one object may be seen as "typed" words, in the sense that concatenation (= composition) is only a partial operation that combines a word $u : A \to B$ with a word of compatible type $v : B \to C$ to produce a word

$uv : A \to C$. For example, consider the graph

$$\mathbb{B}_\Sigma^{\hat{\$}} \quad := \quad \bot \xrightarrow{\hat{}} * \xrightarrow{\$} \top \qquad (* \xrightarrow{a \in \Sigma} *)$$

obtained by extending the bouquet graph $\mathbb{B}_\Sigma$ into a "cylinder". Arrows $w : * \to *$ of $\mathcal{F}\ \mathbb{B}_\Sigma^{\hat{\$}}$ correspond to ordinary words, which may be concatenated freely, but arrows $u : \bot \to *$ may only be prepended to the beginning of a word, while arrows $v : * \to \top$ may only be appended to the end of a word. As we will see in examples later on, it becomes interesting to consider the spliced arrow operad over $\mathcal{F}\ \mathbb{B}_\Sigma^{\hat{\$}}$, which includes operations of the form $f = w_0 - \ldots - w_n\$ : (*,*), \ldots, (*,*) \to (*,\top)$ and $g = \hat{}w_0 - \ldots - w_n : (*,*), \ldots, (*,*) \to (\bot,*)$ that may be seen as spliced words with explicit "end of input" and "beginning of input" markers.

**Remark 1.5.** The operad $\mathcal{W}\,\mathbf{1}$ of spliced arrows over the terminal category is isomorphic to the terminal operad, with a single color $(*,*)$, and a single $n$-ary operation $id - \ldots - id : (*,*), \ldots, (*,*) \to (*,*)$ of every arity $n$. Likewise, the operad of spliced arrows over the product of two categories decomposes as a product of spliced arrow operads $\mathcal{W}\,\mathcal{C} \times \mathcal{D} \cong \mathcal{W}\,\mathcal{C} \times \mathcal{W}\,\mathcal{D}$. This might suggest that the functor $\mathcal{W} - : \mathrm{Cat} \to \mathrm{Oper}$ is a right adjoint, and we will see in §3.2 that this is indeed the case.

1.3. **Context-free grammars over a category.** Classically, a context-free grammar is defined as a tuple $G = (\Sigma, N, S, P)$ consisting of a finite set $\Sigma$ of terminal symbols, a finite set $N$ of non-terminal symbols, a distinguished non-terminal $S \in N$ called the start symbol, and a finite set $P$ of production rules of the form $R \to \sigma$ where $R \in N$ and $\sigma \in (N \cup \Sigma)^*$ is a string of terminal or non-terminal symbols. Observe that any sequence $\sigma$ on the right-hand side of a production can be factored as $\sigma = w_0 R_1 w_1 \ldots R_n w_n$ where $w_0, \ldots, w_n$ are words of terminals and $R_1, \ldots, R_n$ are non-terminal symbols, thus yielding

$$R \to w_0 R_1 w_1 \ldots R_n w_n$$

as the general form of a production rule. We can use this simple observation in order to capture derivations in context-free grammars by functors $p : \mathcal{F}\ \mathbb{S} \to \mathcal{W}\,\Sigma$, as illustrated in Figure 2. More generally, we can use this observation to define context-free grammars over arbitrary categories.

**Definition 1.6.** A **context-free grammar over a category** (or *categorical CFG*) is a tuple $G = (\mathcal{C}, \mathbb{S}, S, p)$ consisting of a category $\mathcal{C}$, a finite species $\mathbb{S}$ equipped with a distinguished color $S \in \mathbb{S}$ called the *start symbol*, and a functor of operads $p : \mathcal{F}\ \mathbb{S} \to \mathcal{W}\,\mathcal{C}$. We then refer to the colors of $\mathbb{S}$ as **non-terminals** and to the operations of $\mathcal{F}\ \mathbb{S}$ as **derivations** or **parse trees**. The **context-free language of arrows** $\mathcal{L}_G$ generated by the grammar $G$ is the subset of arrows in $\mathcal{C}$ which, seen as constants of $\mathcal{W}\,\mathcal{C}$, are in the image of a closed derivation of $S$, that is, $\mathcal{L}_G = \{\, p(\alpha) \mid \alpha : S \,\} \subseteq \mathcal{C}(A, B)$, where $p(S) = (A, B)$.

As suggested in the Introduction, every context-free grammar in the classical sense $G = (\Sigma, N, S, P)$ corresponds to a context-free grammar over $\mathcal{F}\ \mathbb{B}_\Sigma$. For the grammar of Figure 2,

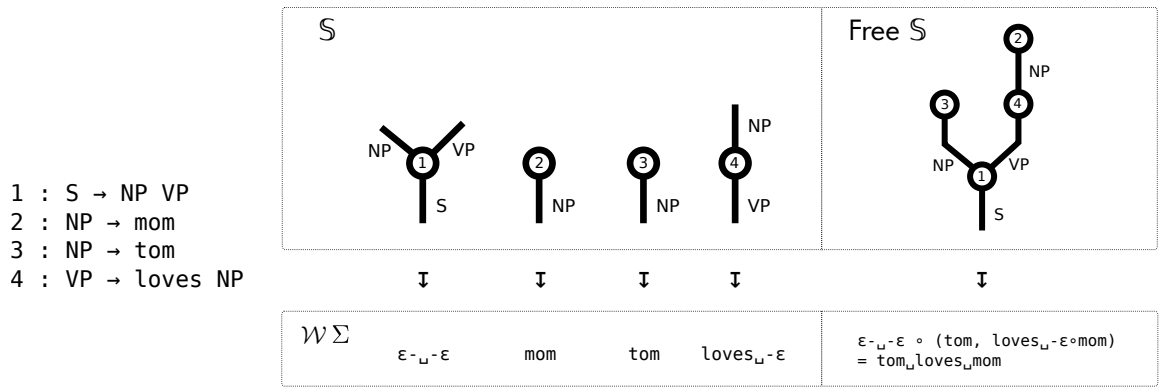



Figure 2: Example of a context-free grammar represented by a functor $\mathcal{F}\ \mathbb{S} \to \mathcal{W}\,\Sigma$, where we have indicated the action of the functor on the generators as well as the induced action on a closed derivation tree.

the corresponding species $\mathbb{S}$ has three colors $\mathsf{NP}, \mathsf{VP}, \mathsf{S}$ and four nodes,

$$\begin{aligned} x_1 &: \mathsf{NP}, \mathsf{VP} \to \mathsf{S} \\ x_2, x_3 &: \mathsf{NP} \\ x_4 &: \mathsf{NP} \to \mathsf{VP} \end{aligned}$$

with the functor $p : \mathcal{F}\ \mathbb{S} \to \mathcal{W}\,\Sigma$ $(= \mathcal{W}\,\mathcal{F}\ \mathbb{B}_\Sigma)$ uniquely defined by the action

$$\begin{aligned} x_1 &\mapsto \varepsilon - \_ - \varepsilon \\ x_2 &\mapsto \mathsf{mom} \\ x_3 &\mapsto \mathsf{tom} \\ x_4 &\mapsto \mathsf{loves}\_ - \varepsilon \end{aligned}$$

on the generators as displayed in the middle of the figure. Conversely, any finite species $\mathbb{S}$ equipped with a color $S \in \mathbb{S}$ and a functor of operads $p : \mathcal{F}\ \mathbb{S} \to \mathcal{W}\,\Sigma$ uniquely determines a context-free grammar over the alphabet $\Sigma$. Indeed, the colors of $\mathbb{S}$ give the non-terminals of the grammar and $S$ the distinguished start symbol, while the nodes of $\mathbb{S}$ together with the functor $p$ give the production rules of the grammar, with each node $x : R_1, \ldots, R_n \to R$ such that $p(x) = w_0 - \ldots - w_n$ determining a context-free production rule $x : R \to w_0 R_1 w_1 \ldots R_n w_n$.

**Proposition 1.7.** *A language $\mathcal{L} \subseteq \Sigma^*$ is context-free in the classical sense if and only if it is the language of arrows of a context-free grammar over $\mathcal{F}\ \mathbb{B}_\Sigma$.*

An interesting feature of the general notion of categorical context-free grammar is that the non-terminals of the grammar $G = (\mathcal{C}, \mathbb{S}, S, p)$ are *sorted* in the sense that every color of $\mathbb{S}$ is mapped by $p$ to a unique color of $\mathcal{W}\,\mathcal{C}$, corresponding to a pair of objects of $\mathcal{C}$. Adapting the conventions from our work on type refinement systems, we sometimes write $R \sqsubset^p (A, B)$ or simply $R \sqsubset (A, B)$ to indicate that $p(R) = (A, B)$ and say that the non-terminal $R$ refines the "gap type" $(A, B)$. We emphasize that the language $\mathcal{L}_G$ generated by a grammar with start symbol $S \sqsubset (A, B)$ is a subset of the homset $\mathcal{C}(A, B)$.

**Example 1.8.** To illustrate some of the versatility afforded by the more general notion of context-free grammar, consider a CFG over the category $\mathcal{F}\ \hat{\mathbb{B}}_\Sigma^{\$}$ from Example 1.4. Such a

grammar may include production rules that can only be applied upon reaching the end of the input or only at the beginning, which is useful in practice, albeit usually modelled in an ad hoc fashion. For example, the grammar of arithmetic expressions defined by Knuth in the original paper on LR parsing [Knu65, example (27)] may be naturally described as a grammar over $\mathcal{F}\ \hat{\mathbb{B}_\Sigma^{\$}}$, which in addition to having three "classical" non-terminals $E, T, P \sqsubset (*,*)$ contains a distinguished non-terminal $S \sqsubset (*,\top)$. Knuth's "zeroth" production rule $0 : S \to E\$$ is then just a unary node $0 : E \to S$ in $\mathbb{S}$, mapped by $p$ to the operation $\varepsilon{-}\$ : (*,*) \to (*,\top)$ in $\mathcal{W}\mathcal{F}\ \hat{\mathbb{B}_\Sigma^{\$}}$.

More significant examples of context-free languages of arrows over categories with more than one object will be given in §3, including context-free grammars over the runs of finite-state automata.

Lastly, let us remark that it is at times useful to consider a categorical CFG equivalently as a triple of a *pointed* finite species $(\mathbb{S}, S)$, a *bipointed* category $(\mathcal{C}, A, B)$, and a functor of pointed operads $p_G : (\mathcal{F}\ \mathbb{S}, S) \to (\mathcal{W}\mathcal{C}, (A,B))$. Note that the operad of spliced arrows construction lifts to a functor

$$\mathcal{W}_\bullet : \mathrm{Cat}_{\bullet,\bullet} \to \mathrm{Oper}_\bullet$$

sending a category $\mathcal{C}$ equipped with a pair of objects $A$ and $B$ to the operad $\mathcal{W}\mathcal{C}$ equipped with the color $(A,B)$, and that the free / forgetful adjunction (1.1) likewise lifts to an adjunction

$$\mathrm{Spec}_\bullet \underset{\mathbb{U}_\bullet}{\overset{\mathcal{F}_\bullet}{\underset{\longleftarrow}{\overset{\longrightarrow}{\bot}}}} \mathrm{Oper}_\bullet$$

between pointed species and pointed operads. This permits another way of understanding the language of arrows generated by a grammar considered as a functor of pointed operads $p_G : \mathcal{F}_\bullet(\mathbb{S}, S) \to \mathcal{W}_\bullet(\mathcal{C}, A, B)$: since the set of constants of color $(A,B)$ in $\mathcal{W}\mathcal{C}$ is in bijection with the set of arrows $A \to B$ in $\mathcal{C}$, we have a natural isomorphism $\mathrm{con} \circ \mathcal{W}_\bullet \cong \hom$ for the evident functors $\hom : \mathrm{Cat}_{\bullet,\bullet} \to \mathrm{Set}$ and $\mathrm{con} : \mathrm{Oper}_\bullet \to \mathrm{Set}$, and $\mathcal{L}_G$ is precisely the image of the function $\mathrm{con}(p_G) : \mathrm{con}(\mathcal{F}\ \mathbb{S}, S) \to \mathrm{con}(\mathcal{W}\mathcal{C}, (A,B)) \cong \mathcal{C}(A,B)$.[3]

1.4. **Properties of a context-free grammar and its associated language.** Standard properties of context-free grammars (cf. [SS88, Ch. 4]), considered as categorical CFGs $G = (\mathcal{F}\ \mathbb{B}_\Sigma, \mathbb{S}, S, p)$, may be reformulated as properties of either the species $\mathbb{S}$, the operad $\mathcal{F}\ \mathbb{S}$, or the functor $p : \mathcal{F}\ \mathbb{S} \to \mathcal{W}\,\Sigma$, with varying degrees of naturality:

- $G$ is *linear* just in case $\mathbb{S}$ only has nodes of arity $\leq 1$. It is *left-linear* (respectively, *right-linear*) just in case it is linear and every unary node $x$ of $\mathbb{S}$ is mapped by $p$ to an operation of the form $\varepsilon{-}w$ (resp. $p(x) = w{-}\varepsilon$).
- $G$ is in *Chomsky normal form* if $\mathbb{S}$ only has nodes of arity 2 or 0, the color $S$ does not appear as the input of any node, every binary node is mapped by $p$ to $\varepsilon{-}\varepsilon{-}\varepsilon$ in $\mathcal{W}\mathcal{F}\ \mathbb{B}_\Sigma$, and every nullary node is mapped to a letter $a \in \Sigma$, unless $R = S$ in which case it is possible that $p(x) = \varepsilon$. (These conditions can be made a bit more more natural by considering $G$ as a context-free grammar over $\mathcal{F}\ \hat{\mathbb{B}_\Sigma^{\$}}$ with $S \sqsubset (*,\top)$, see Example 1.8 above.)
- $G$ is *bilinear* (a generalization of Chomsky normal form [LL09, Lee89]) iff $\mathbb{S}$ only has nodes of arity $\leq 2$.

[3]The reader may refer to the Addendum for a deeper analysis of the language generated by a CFG.

- $G$ is *unambiguous* iff for any pair of constants $\alpha, \beta : S$ in $\mathcal{F}\ \mathbb{S}$, if $p(\alpha) = p(\beta)$ then $\alpha = \beta$. Note that if $p$ is faithful then $G$ is unambiguous, although faithfulness is a stronger condition in general.
- A non-terminal $R$ of $G$ is *nullable* if there exists a constant $\alpha : R$ of $\mathcal{F}\ \mathbb{S}$ such that $p(\alpha) = \varepsilon$.
- A non-terminal $R$ of $G$ is *useful* if there exists a pair of a constant $\alpha : R$ and a unary operation $\beta : R \to S$. Note that if $G$ has no useless non-terminals then $G$ is unambiguous iff $p$ is faithful.

Observe that almost all of these properties can be immediately translated to express properties of context-free grammars over *any* category $\mathcal{C}$. Basic closure properties [HMU07, §7.3] of classical context-free languages (CFLs) also generalize easily to CFLs of arrows.

**Proposition 1.9.**

(1) *If $\mathcal{L}_1, \ldots, \mathcal{L}_k \subseteq \mathcal{C}(A, B)$ are CFLs of arrows, so is their union $\bigcup_{i=1}^{k} \mathcal{L}_i \subseteq \mathcal{C}(A, B)$.*
(2) *If $\mathcal{L}_1 \subseteq \mathcal{C}(A_1, B_1), \ldots, \mathcal{L}_n \subseteq \mathcal{C}(A_n, B_n)$ are CFLs of arrows, and if $w_0 - \ldots - w_n : (A_1, B_1), \ldots, (A_n, B_n) \to (A, B)$ is an operation of $\mathcal{W}\mathcal{C}$, then the "spliced concatenation" $w_0 \mathcal{L}_1 w_1 \ldots \mathcal{L}_n w_n = \{\, w_0 u_1 w_1 \ldots u_n w_n \mid u_1 \in \mathcal{L}_1, \ldots, u_n \in \mathcal{L}_n \,\} \subseteq \mathcal{C}(A, B)$ is also context-free.*
(3) *If $\mathcal{L} \subseteq \mathcal{C}(A, B)$ is a CFL of arrows in a category $\mathcal{C}$ and $F : \mathcal{C} \to \mathcal{D}$ is a functor of categories, then the functorial image $F(\mathcal{L}) \subseteq \mathcal{D}(F(A), F(B))$ is also context-free.*

*Proof.* The proofs of (1) and (2) are just refinements of the standard proofs for context-free languages of words, keeping track of the underlying gap types:

(1) Given grammars $G_i = (\mathcal{C}, \mathbb{S}_i, S_i, p_i)$ for $i = 1 \ldots k$, where all of the $S_i$ refine the same gap type $(A, B)$, we define a new grammar $G = (\mathcal{C}, \mathbb{S}, S, p)$ that generates the union of the languages $\mathcal{L}_G = \bigcup_{i=1}^{k} \mathcal{L}_{G_i}$ by taking $\mathbb{S}$ to be the disjoint union of the colors and nodes of $\mathbb{S}_1, \ldots, \mathbb{S}_k$ combined with a distinguished color $S$ and unary nodes $x_i : S_i \to S$ for $i = 1 \ldots k$, and defining $\phi : \mathbb{S} \to \mathcal{W}\mathcal{C}$ to be the cotupling of $\phi_1, \ldots, \phi_k$ extended by the mappings $S \mapsto (A, B)$ and $x_i \mapsto id_A - id_B$ for $i = 1 \ldots k$.
(2) Given grammars $G_1 = (\mathcal{C}, \mathbb{S}_1, S_1, p_1)$, $\ldots$, $G_n = (\mathcal{C}, \mathbb{S}_n, S_n, p_n)$ where $S_i \sqsubset (A_i, B_i)$ for each $1 \leq i \leq n$, together with an operation $w_0 - \ldots - w_n : (A_1, B_1), \ldots, (A_n, B_n) \to (A, B)$ of $\mathcal{W}\mathcal{C}$, we construct a new grammar $G = (\mathcal{C}, \mathbb{S}, S, p)$ that generates the spliced concatenation $w_0 \mathcal{L}_{G_1} w_1 \ldots \mathcal{L}_{G_n} w_n$ by taking $\mathbb{S}$ to be the disjoint union of the colors and nodes of $\mathbb{S}_1, \ldots, \mathbb{S}_n$ combined with a distinguished color $S$ and a single $n$-ary node $x : S_1, \ldots, S_n \to S$, and defining $\phi : \mathbb{S} \to \mathcal{W}\mathcal{C}$ to be the cotupling of $\phi_1, \ldots, \phi_n$ extended by the mappings $S \mapsto (A, B)$ and $x \mapsto w_0 - \ldots - w_n$.

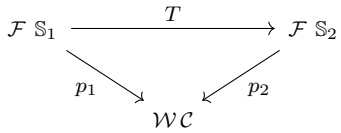
For (3), suppose given a grammar $G = (\mathcal{C}, \mathbb{S}, S, p)$ and a functor of categories $F : \mathcal{C} \to \mathcal{D}$. Then the grammar $F(G)$ generating the language $F(\mathcal{L}_G)$ is defined by postcomposing $p$ with $\mathcal{W}F : \mathcal{W}\mathcal{C} \to \mathcal{W}\mathcal{D}$ while keeping the species $\mathbb{S}$ and start symbol $S$ the same, $F(G) = (\mathcal{D}, \mathbb{S}, S, p\mathcal{W}F)$.

□

Finally, we note that there is a natural definition of morphisms between context-free grammars over the same category, which we refer to as *translations*.

**Definition 1.10.** Let $G_1 = (\mathcal{C}, \mathbb{S}_1, S_1, p_1)$ and $G_2 = (\mathcal{C}, \mathbb{S}_2, S_2, p_2)$ be two grammars over the same category. A **translation** $T : G_1 \to G_2$ is given by a functor of operads $T : \mathcal{F}\ \mathbb{S}_1 \to \mathcal{F}\ \mathbb{S}_2$

that commutes with the projection functors




and preserves the start symbol $T(S_1) = S_2$. We say $T$ is a **strong translation** if the underlying functor $T : \mathcal{F}\ \mathbb{S}_1 \to \mathcal{F}\ \mathbb{S}_2$ is fully faithful, and then write $T : G_1 \hookrightarrow G_2$.

**Proposition 1.11.** *If $T : G_1 \to G_2$ is a translation then $\mathcal{L}_{G_1} \subseteq \mathcal{L}_{G_2}$. If $T : G_1 \hookrightarrow G_2$ is a strong translation, then $\mathcal{L}_{G_1} = \mathcal{L}_{G_2}$, moreover with the grammars generating isomorphic sets of parse trees for the arrows in the language.*

*Proof.* A translation induces a function from constants $\alpha : S_1$ of $\mathcal{F}\ \mathbb{S}_1$ to constants $\beta = T(\alpha) : S_2$ of $\mathcal{F}\ \mathbb{S}_2$ lying over the same arrow $p_1(\alpha) = p_2(\beta)$, and hence $\mathcal{L}_{G_1} \subseteq \mathcal{L}_{G_2}$. If the translation is strong (i.e., $T$ is fully faithful), then this function is invertible, hence $\mathcal{L}_{G_1} = \mathcal{L}_{G_2}$, and the sets of derivations of any given word are isomorphic. □

We will refer to the halves of the proposition as the *weak translation principle* and the *strong translation principle*.

1.5. **A fibrational view of parsing as a lifting problem.** We have seen how any context-free grammar $G = (\mathcal{C}, \mathbb{S}, S, p)$ gives rise to a language $\mathcal{L}_G = \{\, p(\alpha) \mid \alpha : S \,\}$, corresponding to the arrows of $\mathcal{C}$ which, seen as constants of $\mathcal{W}\mathcal{C}$, are in the image of some constant of color $S$ of the free operad $\mathcal{F}\ \mathbb{S}$. However, beyond characterizing the language defined by a grammar, in practice one is often confronted with a dual problem, namely that of parsing: given a word $w$, we want to compute the set of all its parse trees, or at least determine all of the non-terminals which derive it. In our functorial formulation of context-free derivations, this amounts to computing the inverse image of $w$ along the functor $p$, i.e., the set of constants $p^{-1}(w) = \{\, \alpha \mid p(\alpha) = w \,\}$, or alternatively the set of colors in the image of $p^{-1}(w)$ along the output-color function.

To better understand this view of parsing as a lifting problem along a functor of operads, we find it helpful to first recall the correspondence between functors of categories $p : \mathcal{D} \to \mathcal{C}$ and lax functors $F : \mathcal{C} \to \mathrm{Span}(\mathrm{Set})$, where $\mathrm{Span}(\mathrm{Set})$ is the bicategory whose objects are sets, whose 1-cells $S : X \nrightarrow Y$ are spans $X \leftarrow S \to Y$, and whose 2-cells are morphisms of spans. Suppose given such a functor $p : \mathcal{D} \to \mathcal{C}$. To every object $A$ of $\mathcal{C}$ there is an associated "fiber" $F(A) = p^{-1}(A)$ of objects in $\mathcal{D}$ living over $A$, while to every arrow $w : A \to B$ of $\mathcal{C}$ there is an associated fiber $F(w) = p^{-1}(w)$ of arrows in $\mathcal{D}$ living over $w$, equipped with a pair of projection functions $F(A) \leftarrow F(w) \to F(B)$ mapping any lifting $\alpha : R \to S$ of $w : A \to B$ to its source $R \in F(A)$ and target $S \in F(B)$. Moreover, given a pair of composable arrows $u : A \to B$ and $v : B \to C$ in $\mathcal{C}$, there is a morphism of spans

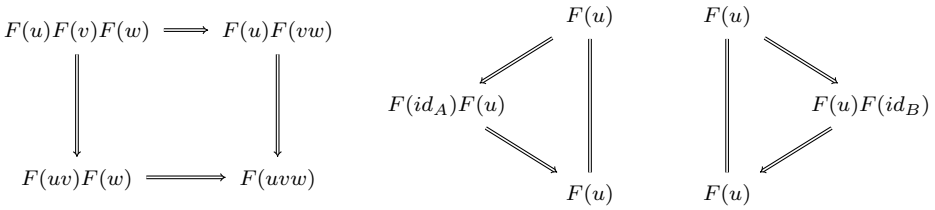
$$F(u)F(v) \Longrightarrow F(uv) \quad : \quad F(A) \nrightarrow F(C) \tag{1.1}$$

from the composite of the spans $F(u) : F(A) \nrightarrow F(B)$ and $F(v) : F(B) \nrightarrow F(C)$ associated to $u : A \to B$ and $v : B \to C$ to the span $F(uv) : F(A) \nrightarrow F(C)$ associated to the composite arrow $uv : A \to C$. This morphism of spans is realized using composition in the category $\mathcal{D}$, namely by the function taking any pair of a lifting $\alpha : R \to S$ of $u$ and a lifting $\beta : S \to T$ of $v$ to the composite $\alpha\beta : R \to T$, which is a lifting of $uv$ by functoriality $p(\alpha\beta) = p(\alpha)p(\beta) = uv$.

Similarly, the identity arrows in the category $\mathcal{D}$ define, for every object $A$ of the category $\mathcal{C}$, a morphism of spans

$$id_{F(A)} \Longrightarrow F(id_A) \quad : \quad F(A) \longrightarrow\!\!\!\!\!+ F(A) \tag{1.2}$$

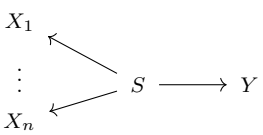
from the identity span $F(A) \leftarrow F(A) \rightarrow F(A)$ to the span associated to the identity arrow $id_A : A \to A$. Associativity and neutrality of composition in $\mathcal{D}$ ensure that the 2-cells (1.1) and (1.2) make the diagrams below commute:




for all triples of composable arrows $u : A \to B$, $v : B \to C$ and $w : C \to D$, and therefore that this collection of data defines what is called a lax functor $F : \mathcal{C} \to \mathrm{Span}(\mathrm{Set})$. In general it is *only* lax, in the sense that the 2-cells $F(u)F(v) \Rightarrow F(uv)$ and $id_{F(A)} \Rightarrow F(id_A)$ are not necessarily invertible.

Conversely, starting from the data provided by a lax functor $F : \mathcal{C} \to \mathrm{Span}(\mathrm{Set})$, we can define a category noted $\int F$ together with a functor $\pi : \int F \to \mathcal{C}$. The category $\int F$ has objects the pairs $(A, R)$ of an object $A$ in $\mathcal{C}$ and an element $R \in F(A)$, and arrows $(w, \alpha) : (A, R) \to (B, S)$ the pairs of an arrow $w : A \to B$ in $\mathcal{C}$ and an element $\alpha \in F(w)$ mapped to $R \in F(A)$ and $S \in F(B)$ by the respective legs of the span $F(A) \leftarrow F(w) \rightarrow F(B)$. The composition and identity of the category $\int F$ are then given by the morphisms of spans $F(u)F(v) \Rightarrow F(uv)$ and $id_{F(A)} \Rightarrow F(id_A)$ witnessing the lax functoriality of $F : \mathcal{C} \to \mathrm{Span}(\mathrm{Set})$. The functor $\pi : \int F \to \mathcal{C}$ is given by the first projection. This construction of a category $\int F$ equipped with a functor $\pi : \int F \to \mathcal{C}$ starting from a lax functor $F : \mathcal{C} \to \mathrm{Span}(\mathrm{Set})$ is a mild variation of Bénabou's construction of the same starting from a lax normal functor $F : \mathcal{C}^{\mathrm{op}} \to \mathrm{Dist}$ [B́00, §7], which is itself a generalization of the well-known Grothendieck construction of a fibration starting from a pseudofunctor $F : \mathcal{C}^{\mathrm{op}} \to \mathrm{Cat}$. One can show that given a functor of categories $p : \mathcal{D} \to \mathcal{C}$, the construction applied to the associated lax functor $F : \mathcal{C} \to \mathrm{Span}(\mathrm{Set})$ induces a category $\int F$ isomorphic to $\mathcal{D}$, in such a way that $p$ coincides with the isomorphism composed with $\pi$. Recently, Ahrens and Lumsdaine [AL19] have introduced the useful terminology "displayed category" to refer to this way of presenting a category $\mathcal{D}$ equipped with a functor $\mathcal{D} \to \mathcal{C}$ as a lax functor $\mathcal{C} \to \mathrm{Span}(\mathrm{Set})$, with their motivations coming from formalization of mathematics.

The constructions which turn a functor of categories $p : \mathcal{D} \to \mathcal{C}$ into a lax functor $F : \mathcal{C} \to \mathrm{Span}(\mathrm{Set})$ and back into a functor $\pi : \int F \to \mathcal{C}$ can be adapted smoothly to functors of operads, viewing $\mathrm{Span}(\mathrm{Set})$ as a 2-categorical operad whose $n$-ary operations $S : X_1, \ldots, X_n \longrightarrow\!\!\!\!\!+ Y$ are multi-legged-spans




or equivalently spans $X_1 \times \cdots \times X_n \leftarrow S \rightarrow Y$, and with the same notion of 2-cell. We will follow Ahrens and Lumsdaine's suggestion and refer to the data of such a lax functor

$F : \mathcal{O} \to \mathrm{Span}(\mathrm{Set})$ representing an operad $\mathcal{D} \cong \int F$ equipped with a functor $p : \mathcal{D} \to \mathcal{O}$ as a **displayed operad**.

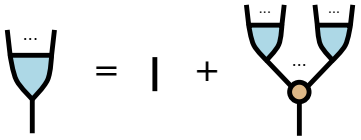
1.6. **An inductive formula and a sequent calculus for displayed free operads.** It is folklore that the free operad over a species $\mathbb{S} = (C, V, i, o)$ may be described concretely as a certain family of trees: operations of $\mathcal{F}\ \mathbb{S}$ are interpreted as (rooted planar) trees whose edges are colored by the elements of $C$ and whose nodes are labelled by the elements of $V$, subject to the constraints imposed by the functions $i : V \to C^*$ and $o : V \to C$. The formal construction of the free operad may be viewed as a free monoid construction, adapted to a situation where the ambient monoidal product (in this case, the composition product of species) is only distributive on the left, see [MSS02, II.1.9] and [BJT97, Appendix B].

From the perspective of programming semantics, it is natural to consider the underlying species of $\mathcal{F}\ \mathbb{S}$ as an inductive data type, corresponding to the initial algebra for the endofunctor $W_{\mathbb{S}}$ on $C$-colored species defined by

$$W_{\mathbb{S}} = \mathbb{R} \mapsto \mathbb{I} + \mathbb{S} \circ \mathbb{R}$$

where $+$ denotes the coproduct of $C$-colored species which is constructed by taking the disjoint union of nodes, while $\circ$ and $\mathbb{I}$ denote respectively the composition product of $C$-colored species and the identity species, defined as follows. Given two $C$-colored species $\mathbb{S}$ and $\mathbb{R}$, the $n$-ary nodes $R_1, \ldots, R_n \to R$ of $\mathbb{S} \circ \mathbb{R}$ are formal composites $g \bullet (f_1, \ldots, f_k)$ consisting of a node $g : S_1, \ldots, S_k \to S$ of $\mathbb{S}$ and of a tuple of nodes $f_1 : \Gamma_1 \to S_1$, $\ldots$, $f_k : \Gamma_k \to S_k$ of $\mathbb{R}$, such that the concatenation of the lists of colors $\Gamma_1, \ldots, \Gamma_k$ is equal to the list $R_1, \ldots, R_n$. The unit $\mathbb{I}$ is the $C$-colored species with a single unary node $*_R : R \to R$ for every color $R \in C$, and no other nodes.

As the initial $W_{\mathbb{S}}$-algebra, the free operad over $\mathbb{S}$ is equipped with a map of species $\mathbb{I} + \mathbb{S} \circ \mathcal{F}\ \mathbb{S} \longrightarrow \mathcal{F}\ \mathbb{S}$, which by the Lambek Lemma is invertible, with the following interpretation: any operation of $\mathcal{F}\ \mathbb{S}$ is either an identity operation, or the parallel composition of a node of $\mathbb{S}$ with a list of operations of $\mathcal{F}\ \mathbb{S}$. Note that this interpretation also corresponds to a canonical decomposition



(1.3)

of rooted planar trees, by iterated examination of the root. An operation of $\mathcal{F}\ \mathbb{S}$ may be seen as a rooted planar tree whose edges and nodes are labelled by the species $\mathbb{S}$, and we sometimes refer to such an operation as an $\mathbb{S}$-*rooted tree* (cf. [BLL98, §3.2]).

It is possible to derive an analogous inductive characterization of *functors* $p : \mathcal{F}\ \mathbb{S} \to \mathcal{O}$ from a free operad into an arbitrary operad $\mathcal{O}$ considered as displayed free operads, i.e., as lax functors $F : \mathcal{O} \to \mathrm{Span}(\mathrm{Set})$ generated by an underlying map of species $\phi : \mathbb{S} \to \mathcal{O}$. Two subtleties arise. First, that the species $\mathbb{S}$ and the operad $\mathcal{O}$ may in general have a different set of colors, related by the change-of-color function $\phi_C$. To account for this, rather than restricting the operations $+, \circ, \mathbb{I}$ to the category of $C$-colored species, one should consider them as global functors

$$+, \circ : \mathrm{Spec} \times_{\mathrm{Set}} \mathrm{Spec} \to \mathrm{Spec} \qquad \mathbb{I} : \mathrm{Set} \to \mathrm{Spec}$$

on the "polychromatic" category of species, which respect the underlying sets of colors in a functorial way. Second, and more significantly, the above functor $W_{\mathbb{S}}$ transports a

species $\mathbb{R}$ living over $\mathcal{O}$ to a species living over $\mathbb{I} + \mathcal{O} \circ \mathcal{O}$, so that in order to obtain again a species living over $\mathcal{O}$ (and thus define an endofunctor) one needs to "push forward" along the canonical $W_{\mathcal{O}}$-algebra $[e, m] : \mathbb{I} + \mathcal{O} \circ \mathcal{O} \longrightarrow \mathcal{O}$ that encodes the operad structure of $\mathcal{O}$, seen as a monoid in $(\mathrm{Spec}, \circ, \mathbb{I})$. A detailed discussion is beyond the scope of this paper, but we nevertheless state the following proposition, whose intuitive content should be clear:

**Proposition 1.12.** *Let $\phi : \mathbb{S} \to \mathcal{O}$ be a map of species from a species $\mathbb{S}$ into an operad $\mathcal{O}$, and let $p : \mathcal{F}\,\mathbb{S} \to \mathcal{O}$ be the corresponding functor from the free operad. Then the associated lax functor $F : \mathcal{O} \to \mathrm{Span}(\mathrm{Set})$ computing the fibers of $p$ is given by $F(A) = \phi^{-1}(A)$ on colors of $\mathcal{O}$, and by the least family of spans $F(f)$ indexed by operations $f : A_1, \ldots, A_n \to A$ of $\mathcal{O}$ such that*

$$F(f) \cong \sum_{\substack{f = id_A \\ \phi(R) = A}} id_R + \sum_{f = g \circ (h_1, \ldots, h_k)} \phi^{-1}(g) \bullet (F(h_1), \ldots, F(h_k)) \tag{1.4}$$

*where we write $\circ$ for composition in the operad $\mathcal{O}$ and $\bullet$ for formal composition of nodes in $\mathbb{S}$ with operations in $\mathcal{F}\,\mathbb{S}$. Specializing the formula to constant operations, the left summand disappears and* (1.4) *simplifies to:*

$$F(c) \cong \sum_{c = g \circ (c_1, \ldots, c_k)} \phi^{-1}(g) \bullet (F(c_1), \ldots, F(c_k)) \tag{1.5}$$

The inductive characterization of displayed free operads may be equivalently expressed as a sequent calculus, see Figure 3, which is parameterized by an underlying map of species $\phi : \mathbb{S} \to \mathcal{O}$. Adapting a convention from our type refinement systems work, we write

$$R_1, \ldots, R_k \xRightarrow[g]{\phi} R$$

to indicate the existence of a node $x : R_1, \ldots, R_k \to R$ in $\mathbb{S}$ such that $\phi(x) = g$, and similarly

$$S_1, \ldots, S_n \xRightarrow[f]{p} R$$

for an operation $f : S_1, \ldots, S_n \to R$ in $\mathcal{F}\,\mathbb{S}$ such that $p(\alpha) = f$. Again, we write $R \sqsubset^p A$ to indicate that $R$ is a color of $\mathcal{F}\,\mathbb{S}$ with image $p(R) = A$ in $\mathcal{O}$. As standard, we treat the inference rules as an inductive definition, in the sense that the set of derivations of the calculus is the least set closed under the inference rules. Then formulas (1.4) and (1.5) are equivalent to the following:

**Proposition 1.13.** *For the sequent calculus of Fig. 3, parameterized with respect to a map of species $\phi : \mathbb{S} \to \mathcal{O}$ and the corresponding functor $p : \mathcal{F}\,\mathbb{S} \to \mathcal{O}$, derivations of $\Gamma \Rightarrow^p_f R$ are in one-to-one correspondence with operations $\alpha : \Gamma \to R$ in $\mathcal{F}\,\mathbb{S}$ such that $p(\alpha) = f$. As a special case, constants $\alpha : R$ such that $p(\alpha) = c$ are in one-to-one correspondence with derivations of $\Rightarrow^p_c R$, which necessarily only use instances of the* NODE *rule.*

1.7. **Application to parsing.** Instantiating (1.5) with the underlying functor of a categorical CFG generated by a map of species $\phi : \mathbb{S} \to \mathcal{W}\mathcal{C}$, we immediately obtain the following characteristic formula for the family of sets of parse trees $F(w)$ of an arrow $w$ in $\mathcal{C}$, seen as liftings of the constant $w$ in $\mathcal{W}\mathcal{C}$ to a constant in $\mathcal{F}\,\mathbb{S}$:

$$F(w) \cong \sum_{w = w_0 u_1 w_1 \ldots u_n w_k} \phi^{-1}(w_0 - \ldots - w_k) \bullet (F(u_1), \ldots, F(u_k)) \tag{1.6}$$

$$\textsc{leaf}\ \frac{R \overset{p}{\sqsubset} A}{R \underset{id_A}{\overset{p}{\Longrightarrow}} R} \qquad \textsc{node}\ \frac{f = g \circ (h_1, \dots, h_k) \qquad R_1, \dots, R_k \underset{g}{\overset{\phi}{\Longrightarrow}} R \qquad \Gamma_1 \underset{h_1}{\overset{p}{\Longrightarrow}} R_1 \quad \dots \quad \Gamma_k \underset{h_k}{\overset{p}{\Longrightarrow}} R_k}{\Gamma_1, \dots, \Gamma_k \underset{f}{\overset{p}{\Longrightarrow}} R}$$

Figure 3: A sequent calculus for displayed free operads.

Equivalently, specializing the sequent calculus of Figure 3 and translating the generic judgment forms into traditional CFG notation by

$$\begin{aligned} R_1, \dots, R_k \underset{w_0 - \dots - w_n}{\overset{\phi}{\Longrightarrow}} R \quad &\rightsquigarrow \quad R \to w_0 R_1 w_1 \dots R_n w_n \\ \underset{w}{\overset{p}{\Longrightarrow}} R \quad &\rightsquigarrow \quad R \to^+ w \end{aligned}$$

we obtain the following inference rule:

$$\frac{w = w_0 u_1 w_1 \dots u_k w_k \qquad R \to w_0 R_1 w_1 \dots R_k w_k \qquad R_1 \to^+ u_1 \quad \dots \quad R_k \to^+ u_k}{R \to^+ w} \tag{1.7}$$

inductively defining the set of closed context-free derivations. For any given word $w$, let $N_w = \{\, R \mid R \to^+ w \,\}$ be the set of non-terminals that derive it, corresponding to the image of $F(w)$ along the function returning the root label of a parse tree. We have that:

$$\begin{aligned} R \in N_w \iff w = w_0 u_1 w_1 \dots u_k w_k \wedge \\ R \to w_0 R_1 w_1 \dots R_k w_k \wedge R_1 \in N_{u_1} \wedge \dots \wedge R_k \in N_{u_k} \end{aligned} \tag{1.8}$$

This equation is essentially the characteristic formula expressed by Leermakers [Lee89] for the defining relation of the "C-parser", which generalizes the well-known Cocke-Younger-Kasami (CYK) algorithm. Presentations of the CYK algorithm are usually restricted to grammars in Chomsky normal form (cf. [LL09]), but as observed by Leermakers, the relation $N_w$ defined by (1.8) can be solved effectively for any context-free grammar $G$ and given word $w = a_1 \dots a_n$ by building up a parse matrix $N_{i,j}$ indexed by the subwords $w_{i,j} = a_{i+1} \dots a_j$ for all $1 \leq i \leq j \leq n$, and yields a cubic complexity algorithm in the case that $G$ is bilinear in the sense of §1.4. Moreover, by adding non-terminals, it is always possible to transform a CFG into a bilinear CFG that generates the same language, even preserving the original derivations up to isomorphism.

**Proposition 1.14.** *For any categorical context-free grammar $G = (\mathcal{C}, \mathbb{S}, S, p)$, there is a bilinear CFG $G_{\mathsf{bin}} = (\mathcal{C}, \mathbb{S}_{\mathsf{bin}}, S, p_{\mathsf{bin}})$ together with a fully faithful functor of operads $B : \mathcal{F}\,\mathbb{S} \to \mathcal{F}\,\mathbb{S}_{\mathsf{bin}}$ such that $p = B p_{\mathsf{bin}}$. In particular, by the strong translation principle (Prop. 1.11) we have that $\mathcal{L}_G = \mathcal{L}_{G_{\mathsf{bin}}}$ with isomorphic sets of parse trees.*

*Proof.* Given $G = (\mathcal{C}, \mathbb{S}, S, p)$, a bilinear grammar $G_{\mathsf{bin}} = (\mathcal{C}, \mathbb{S}_{\mathsf{bin}}, S, p_{\mathsf{bin}})$ is constructed as follows. $\mathbb{S}_{\mathsf{bin}}$ includes all of the colors and all of the nullary nodes of $\mathbb{S}$, with $\phi_{\mathsf{bin}}(R) = \phi(R)$ and $\phi_{\mathsf{bin}}(c) = \phi(c)$. Additionally, for every node $x : R_1, \dots, R_n \to R$ of $\mathbb{S}$ of positive arity $n > 0$, where $\phi(x) = w_0 - \dots - w_n : (A_1, B_1), \dots, (A_n, B_n) \to (A, B)$ in $\mathcal{W}\mathcal{C}$, we include:

- $n$ new colors $I_{x,0}, \dots, I_{x,n-1}$, with $\phi_{\mathsf{bin}}(I_{x,i-1}) = (A, A_i)$ for $1 \leq i \leq n$;
- one nullary node $x_0 : I_{x,0}$, with $\phi_{\mathsf{bin}}(x_0) = w_0$;
- $n$ binary nodes $x_1, \dots, x_n$, where $x_i : I_{x,i-1}, R_i \to I_{x,i}$ and $\phi_{\mathsf{bin}}(x_i) = id_A - id_{A_i} - w_i$ for all $1 \leq i \leq n$, under the convention that $I_{x,n} = R$.

We define the functor $B : \mathcal{F}\,\mathbb{S} \to \mathcal{F}\,\mathbb{S}_{\mathsf{bin}}$ on colors by $B(R) = R$, on nullary nodes by $B(c) = c$, and on nodes $x : R_1, \ldots, R_n \to R$ of positive arity by $B(x) = x_n \circ_0 \cdots \circ_0 x_1 \circ_0 x_0$. By induction on $n$, there is a one-to-one correspondence between nodes $x : R_1, \ldots, R_n \to R$ of $\mathbb{S}$ and operations $B(x) : R_1, \ldots, R_n \to R$ of $\mathcal{F}\,\mathbb{S}_{\mathsf{bin}}$, so the functor $B$ is fully faithful. □

1.8. **Generalized context-free grammars.** We conclude this section by observing that for many purposes, a functor of operads from a free operad generated by a finite species $\mathbb{S}$ into an arbitrary operad $\mathcal{O}$ may be considered (after selection of a start symbol in $\mathbb{S}$) as a "generalized" context-free grammar, generating a language of constants in $\mathcal{O}$.

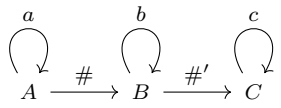
**Definition 1.15.** A **generalized context-free grammar (over an operad)** is a tuple $G = (\mathcal{O}, \mathbb{S}, S, p)$ consisting of an operad $\mathcal{O}$, a finite species $\mathbb{S}$ equipped with a distinguished color $S \in \mathbb{S}$, and a functor of operads $p : \mathcal{F}\,\mathbb{S} \to \mathcal{O}$. The **generalized context-free language of constants** generated by the grammar $G$ is given by the subset of constants (or "conset") $\mathcal{L}_G = \{\, p(\alpha) \mid \alpha : S \,\} \subseteq \mathcal{O}(A)$, where $p(S) = A$.

Hence a classical context-free word grammar can be seen as a generalized CFG (gCFG) over a spliced word operad $\mathcal{O} = \mathcal{W}\,\Sigma$, and a categorical CFG as a gCFG over a spliced arrow operad $\mathcal{O} = \mathcal{W}\,\mathcal{C}$. By selecting $\mathcal{O}$ appropriately, Definition 1.15 encompasses other extensions of the notion of context-free grammar from the literature.

**Example 1.16.** For any operad $\mathcal{P}$, one can construct a new operad $!_{\mathsf{aff}}\mathcal{P}$ whose colors are lists $[A_1, \ldots, A_k]$ of colors in $\mathcal{P}$ and whose $n$-ary operations $[\Gamma_1], \ldots, [\Gamma_n] \to [A_1, \ldots, A_k]$ are pairs $([f_1, \ldots, f_k], \sigma)$ of a list of operations $f_1 : \Omega_1 \to A_1, \ldots, f_k : \Omega_k \to A_k$ in $\mathcal{P}$ and an injection $\sigma : \Omega_1, \ldots, \Omega_k \hookrightarrow \Gamma_1, \ldots, \Gamma_n$ from the concatenation of the inputs of the $k$ operations to the concatenation of the $n$ input lists. Observe that if $\mathcal{P}$ is an un(i)colored operad, then the colors of $!_{\mathsf{aff}}\mathcal{P}$ are isomorphic to natural numbers. Let us also remark that $!_{\mathsf{aff}}\mathcal{P}$ is in fact the *free semi-cartesian (or "affine") strict monoidal operad* over $\mathcal{P}$.[4]

A gCFG over $!_{\mathsf{aff}}\mathcal{W}\,\Sigma$ with start symbol $S \sqsubset 1$ is precisely a *multiple context-free grammar* [SMFK91], while more generally a gCFG over $!_{\mathsf{aff}}\mathcal{W}\,\mathcal{C}$ with $S \sqsubset [(A, B)]$ (for some objects $A$ and $B$ of $\mathcal{C}$) could be called a "multiple categorical CFG". Such a grammar is a *$k$-multiple CFG* (again in the sense of [SMFK91]) just in case every non-terminal $R \sqsubset [\Gamma]$ refines a list of length $|\Gamma| \leq k$.

For instance, let $\mathcal{C}$ be the free category over the following graph:




We can define a 3-mCFG generating the language $a^n \# b^n \#' c^n$ with a pair of non-terminals

$$S \sqsubset [(A, C)]$$
$$R \sqsubset [(A, A), (B, B), (C, C)]$$

and production rules specified by the triple of nodes in $\mathbb{S}$

$$x_1 : R$$
$$x_2 : R \to R$$
$$x_3 : R \to S$$

[4] See [Lei04, p. 56–58] for discussion of the similar constructions of the free PRO(P) over a colored operad.

mapped respectively to the following operations in $!_{\mathsf{aff}}\mathcal{W}\mathcal{C}$

$$[id_A, id_B, id_C] : [(A,A),(B,B),(C,C)]$$
$$[a{-}id_A, b{-}id_B, c{-}id_C] : [(A,A),(B,B),(C,C)] \to [(A,A),(B,B),(C,C)]$$
$$[-\#-\#'-] : [(A,A),(B,B),(C,C)] \to [(A,C)]$$

where we have elided the injection component and only shown the lists of operations in $\mathcal{W}\mathcal{C}$, since the injection $\sigma$ used is always the identity.

**Example 1.17.** The *free cartesian strict monoidal operad* $!_{\mathsf{cart}}\mathcal{P}$ over an operad $\mathcal{P}$ is constructed just as $!_{\mathsf{aff}}\mathcal{P}$, but replacing injections by arbitrary functions $\sigma : \Omega_1, \dots, \Omega_k \to \Gamma_1, \dots, \Gamma_n$. A gCFG over $!_{\mathsf{cart}}\mathcal{W}\,\Sigma$ is precisely a *parallel multiple context-free grammar* in the sense of [SMFK91], and more generally a gCFG over $!_{\mathsf{cart}}\mathcal{W}\mathcal{C}$ could be called a "parallel multiple categorical CFG".

Examples of generalized context-free languages of a more semantic flavor can also be derived by taking $\mathcal{O}$ to be Set or similar. Recall that Set is the (large) operad whose colors are sets and whose operations $f : X_1, \dots, X_n \to X$ are $n$-ary functions $f : X_1 \times \cdots \times X_n \to X$, with composition defined as expected.

**Example 1.18.** (Adapted from [CE12, §1.1.3].) Let $\mathrm{DiGr}_{\bullet,\bullet}$ be the set of finite directed graphs equipped with two distinct vertices labeled $\mathsf{src}$ and $\mathsf{tgt}$. The *parallel composition* $G \parallel H$ of a pair of graphs $(G,H) \in \mathrm{DiGr}_{\bullet,\bullet} \times \mathrm{DiGr}_{\bullet,\bullet}$ is formed by starting from the disjoint union of the two graphs and identifying their two sources and two targets to form a graph $G \parallel H \in \mathrm{DiGr}_{\bullet,\bullet}$ with $\mathsf{src}(G \parallel H) = \mathsf{src}(G) = \mathsf{src}(H)$ and $\mathsf{tgt}(G \parallel H) = \mathsf{tgt}(G) = \mathsf{tgt}(H)$. The *series composition* $G;H$ is formed by similarly starting from the disjoint union of $G$ and $H$ but then identifying $\mathsf{tgt}(G) = \mathsf{src}(H)$ and taking $\mathsf{src}(G;H) = \mathsf{src}(G)$ and $\mathsf{tgt}(G;H) = \mathsf{tgt}(H)$. We can define a gCFG over Set with one non-terminal $S \sqsubset \mathrm{DiGr}_{\bullet,\bullet}$ and three production rules corresponding to a pair of binary nodes $par : S,S \to S$ and $seq : S,S \to S$ mapped to the operations $\parallel$ and ; respectively, plus a constant $e : S$ mapped to the unique directed graph $\bullet \to \bullet$ with one edge and two vertices. The language of constants generated by this gCFG is exactly the set of *series-parallel graphs*.

Many of the properties of categorical CFGs discussed in §1.4 extend directly to gCFGs, as do all of the closure properties of CFLs and the translation principles. For the record, we state here the appropriate generalizations for gCFLs of the closure properties listed in Proposition 1.9, omitting the proofs since they are essentially identical.

**Proposition 1.19.**

(1) *If $\mathcal{L}_1, \dots, \mathcal{L}_k \subseteq \mathcal{O}(A)$ are gCFLs, so is their union $\bigcup_{i=1}^k \mathcal{L}_i \subseteq \mathcal{O}(A)$.*
(2) *If $\mathcal{L}_1 \subseteq \mathcal{O}(A_1), \dots, \mathcal{L}_n \subseteq \mathcal{O}(A_n)$ are gCFLs, and if $f : A_1, \dots, A_n \to A$ is an operation of $\mathcal{O}$, then the application $f(\mathcal{L}_1, \dots, \mathcal{L}_n) = \{\, f(u_1, \dots, u_n) \mid u_1 \in \mathcal{L}_1, \dots, u_n \in \mathcal{L}_n \,\} \subseteq \mathcal{O}(A)$ is also a gCFL.*
(3) *If $\mathcal{L} \subseteq \mathcal{O}(A)$ is a gCFL of constants in an operad $\mathcal{O}$ and $F : \mathcal{O} \to \mathcal{P}$ is a functor of operads, then the functorial image $F(L) \subseteq \mathcal{P}(F(A))$ is also a gCFL.*

1.9. **Related work.** Although the notion of context-free grammar over a category seems to be new, the idea of viewing classical context-free grammars fibrationally and representing them as functors appears in various guises in the literature. Walters briefly described [Wal89] how context-free grammars may be represented by certain morphisms of multigraphs that we would call maps of species (see Footnote 2 above) and might notate $\phi : \mathbb{S} \to \tilde{\Sigma}$, where $\tilde{\Sigma}$ is defined similarly to the underlying species of $\mathcal{W}\,\Sigma$ but with two distinct kinds of nodes: nullary nodes $a : *$ for every letter $a \in \Sigma$, and $n$-ary nodes $\mu_n : *^n \to *$ standing for concatenation. Note that taking $\tilde{\Sigma}$ so-defined as the target of $\phi$ imposes a restriction on the grammar (similar to Chomsky normal form), that all production rules are either of the form $R \to a$ or $R \to R_1 \dots R_n$. In order to recover the language generated by a grammar with start symbol $S$, Walters next considers the functor $\mathcal{F}_\times \phi : \mathcal{F}_\times \mathbb{S} \to \mathcal{F}_\times \tilde{\Sigma}$ between the *free categories with products* generated by the species, then composes with the canonical functor $\psi : \mathcal{F}_\times \tilde{\Sigma} \to \mathrm{Mon}_\Sigma$ into the Lawvere theory of monoids containing $\Sigma$, and finally takes the image $\psi(\mathcal{F}_\times \phi(\mathcal{F}_\times \mathbb{S}(1, S)))$. All in all, we find the encoding of classical CFGs by functors of operads $\mathcal{F}\ \mathbb{S} \to \mathcal{W}\,\Sigma$ to be simpler, more direct and more general, but Walters' approach is certainly similar in spirit.

The methodology of *abstract categorial grammars* introduced by De Groote [dG01] is probably even more closely related to our approach. Based on linear lambda-calculus, the approach consists of expressing different kinds of grammars by specifying a signature $\Sigma_1$ as an "abstract vocabulary", a signature $\Sigma_2$ as an "object vocabulary", and a "lexicon" given by a morphism of signatures $\Sigma_1 \to \Sigma_2$. In particular, context-free grammars are represented by taking the abstract vocabulary $\Sigma_1$ to have a type for each non-terminal and a constant $x : R_1 \multimap \cdots \multimap R_n \multimap R$ for every production rule $x : R \to w_0 R_1 \dots R_n w_n$, the object vocabulary $\Sigma_2$ to have a single type $*$ and a constant $a : * \multimap *$ for every letter, and the lexicon to map every non-terminal $R$ to $* \multimap *$, and every constant $x$ as above to the lambda-term $\lambda u_1 \dots u_n . w_0 \circ u_1 \circ \cdots \circ u_n \circ w_n$. Although the definition of ACGs was originally given type-theoretically rather than categorically, the framework could very clearly be reformulated naturally in an operadic setting, using free closed operads, and seems to have deep connections with the approach described here. One way of making this connection more precise could be to prove that the interpretation $(A, B) \mapsto A \multimap B$ extends to a full and faithful embedding $\mathcal{W}\,\mathcal{C} \to \mathcal{F}_{\multimap}\,\mathcal{C}$ of the operad of spliced arrows $\mathcal{W}\,\mathcal{C}$ into the free closed operad over $\mathcal{C}$, where the operations of $\mathcal{F}_{\multimap}\,\mathcal{C}$ may be represented by $\beta\eta$-normal (ordered) linear lambda terms. It is worth emphasizing that ACGs were introduced to capture a wide variety of grammatical formalisms, including categorial grammars in the style of Lambek calculus but also extensions of context-free grammars (like we discussed in §1.8) such as multiple CFGs [dGP04].

## 2. Finite state automata over categories and operads

We have seen how the classical notion of context-free grammar generalizes naturally to define context-free grammars over any category $\mathcal{C}$, or even over any operad $\mathcal{O}$, presented by a functor of operads $\mathcal{F}\ \mathbb{S} \to \mathcal{W}\,\mathcal{C}$ or $\mathcal{F}\ \mathbb{S} \to \mathcal{O}$ together with a distinguished start symbol in $\mathbb{S}$. In this section, we carry on in the same fibrational vein and begin by explaining how classical non-deterministic finite-state word automata may be generalized to define NDFAs over any category, presented by functors of categories $\mathcal{Q} \to \mathcal{C}$ that are both *finitary* and have the *unique lifting of factorizations* (ULF) property, together with distinguished initial and accepting states in $\mathcal{Q}$. The base category $\mathcal{C}$ is often freely generated, as in the case of

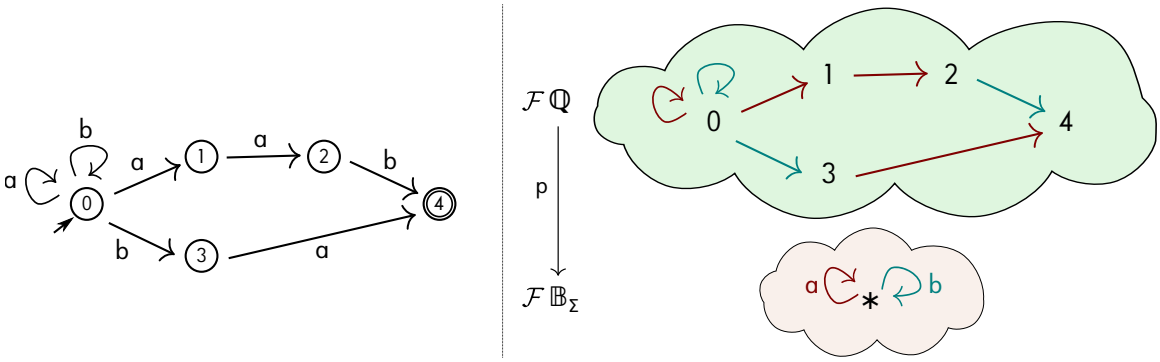
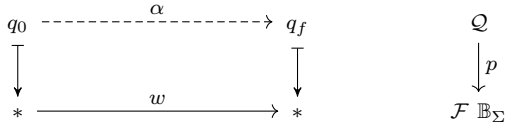



Figure 4: Left: an NDFA represented with a traditional state-diagram. Right: the bare NDFA as a finitary ULF functor $p : \mathcal{F}\ \mathbb{Q} \to \mathcal{F}\ \mathbb{B}_\Sigma$. (We do not label the generators of $\mathcal{F}\ \mathbb{Q}$ or indicate composite arrows, and we use colors to indicate the images of the generators in $\mathcal{F}\ \mathbb{B}_\Sigma$.)

word automata, but we also give natural examples of automata over non-free categories. One important aspect of this approach is that it adapts smoothly when one shifts from word automata to tree automata, simply by replacing categories with operads. Indeed, a key fact is that the splicing construction transports any finitary ULF functor of categories $p : \mathcal{Q} \to \mathcal{C}$ to a finitary ULF functor of operads $\mathcal{W}\,p : \mathcal{W}\,\mathcal{Q} \to \mathcal{W}\,\mathcal{C}$, thereby transforming an NDFA over a category into an NDFA over its operad of spliced arrows. As we will see in the following section (§3), this observation greatly facilitates the proof that context-free languages are closed under intersection with regular languages.

2.1. **Non-deterministic word automata as finitary ULF functors over categories.** Classically, a non-deterministic finite state automaton may be represented by a finite alphabet $\Sigma$, a finite set $Q$ of states, a finite set *Tran* of transitions together with a labelling function $\delta : \mathit{Tran} \to Q \times \Sigma \times Q$, and a choice of initial and accepting states. We will focus first on the underlying "bare" automaton $M = (\Sigma, Q, \delta)$ before the choice of initial and accepting states. Every such bare automaton $M$ induces a functor of categories $p : \mathcal{Q} \to \mathcal{F}\ \mathbb{B}_\Sigma$ where $\mathcal{Q}$ is the category with set of objects $Q$ and with arrows freely generated by arrows of the form $t : q \to q'$ for every transition $t \in \mathit{Tran}$ such that $\delta(t) = (q, a, q')$, and where the functor $p : \mathcal{Q} \to \mathcal{F}\ \mathbb{B}_\Sigma$ transports every such generator $t : q \to q'$ to the arrow $a : * \to *$ representing the letter $a \in \Sigma$ in the category $\mathcal{F}\ \mathbb{B}_\Sigma$. See Figure 4 for an example. Under this formulation, any composite arrow $\alpha : q_0 \to q_f$ of the category $\mathcal{Q}$ describes a run of the automaton $M$ over the word $w = p(\alpha) : * \to *$ which starts in state $q_0 \in Q$ and ends in state $q_f \in Q$, as depicted below:




One distinctive property of the functor $p : \mathcal{Q} \to \mathcal{F}\ \mathbb{B}_\Sigma$ is that it has the *unique lifting of factorizations* (ULF) property [Law86, BN00, BF00, LM10]. Recall that a functor of categories has the ULF property (or "is ULF") when any factorization of the image of an arrow lifts uniquely to a factorization of that arrow.

**Definition 2.1.** A functor $p : \mathcal{D} \to \mathcal{C}$ is said to be **ULF** if for every arrow $\alpha$ of $\mathcal{D}$, if $p(\alpha) = uv$ for some pair of arrows $u$ and $v$ of $\mathcal{C}$, there exists a unique pair of arrows $\beta$ and $\gamma$ in $\mathcal{D}$ such that $\alpha = \beta\gamma$ and $p(\beta) = u$ and $p(\gamma) = v$.

Such functors are also known as *discrete Conduché fibrations* [Joh99, Gue20], and they provide a common generalization of both discrete fibrations and discrete opfibrations. The ULF condition arises naturally when considering the correspondence recalled in §1.5 between functors $\mathcal{D} \to \mathcal{C}$ and lax functors $\mathcal{C} \to \mathrm{Span}(\mathrm{Set})$.

**Proposition 2.2.** *A functor of categories $p : \mathcal{D} \to \mathcal{C}$ is ULF iff the associated lax functor $F : \mathcal{C} \to \mathrm{Span}(\mathrm{Set})$ is a pseudofunctor.*

*Proof.* The definition of ULF says precisely that the structure maps $F(u)F(v) \Longrightarrow F(uv)$ of the lax fiber functor are invertible. Moreover, it is an immediate consequence of the definition that $p(\alpha) = id_A$ implies $\alpha$ is an identity arrow, since otherwise there would be two distinct liftings $\alpha = id\,\alpha$ and $\alpha = \alpha\, id$ of the factorization $id_A = id_A\, id_A$. Hence the structure maps $id_{F(A)} \Longrightarrow F(id_A)$ are also invertible, and $F$ is a pseudofunctor. □

The ULF property reflects an important structural property of non-deterministic finite state automata: that every arrow $\alpha : q_0 \to q_f$ lying above some arrow $p(\alpha) = w$ corresponding to a run of the automaton can be factored uniquely as a sequence of transitions along the letters of the word $w$. In the case of the ULF functor $p : \mathcal{Q} \to \mathcal{F}\ \mathbb{B}_\Sigma$ that we construct from a classical NDFA, both the categories $\mathcal{Q}$ and $\mathcal{F}\ \mathbb{B}_\Sigma$ are freely generated, and the functor $p$ is generated from a homomorphism of graphs. (In effect, on the right-hand side of Figure 4 we visualize the functor by this generating graph homomorphism.) This is not a coincidence, as *any* ULF functor into a free category is necessarily of this form.

**Proposition 2.3** ([Str96], [Gue20])**.** *Let $p : \mathcal{D} \to \mathcal{C}$ be a functor into a category $\mathcal{C} = \mathcal{F}\ \mathbb{G}$ freely generated by some graph $\mathbb{G}$. Then $p$ is ULF iff $\mathcal{D} = \mathcal{F}\ \mathbb{H}$ is free over some graph $\mathbb{H}$ and $p = \mathcal{F}\ \phi$ is generated by a graph homomorphism $\phi : \mathbb{H} \to \mathbb{G}$.*

*Proof.* The right-to-left direction is immediate. For the left-to-right implication, we take $\mathbb{H} = p^{-1}(\mathbb{G})$ to be the graph containing just those arrows of $\mathcal{D}$ lying over the generators of $\mathcal{C}$. Since the image of any arrow $\alpha$ in $\mathcal{D}$ uniquely decomposes as a composition of edges $p(\alpha) = e_1 \cdots e_n$ in $\mathbb{G}$, the ULF property implies that $\alpha$ uniquely decomposes as a composition of edges in $\mathbb{H}$, and $p$ is generated by its restriction to the generators $\phi : \mathbb{H} \to \mathbb{G}$. □

Of course another important property of NDFAs is their finite nature, which may also be expressed as a property of the corresponding functor $p : \mathcal{Q} \to \mathcal{F}\ \mathbb{B}_\Sigma$.

**Definition 2.4.** We say that a functor $p : \mathcal{D} \to \mathcal{C}$ is **finitary** if either of the following equivalent conditions hold:

- the fiber $p^{-1}(A)$ as well as the fiber $p^{-1}(w)$ is finite for every object $A$ and arrow $w$ in the category $\mathcal{C}$;
- the associated lax functor $F : \mathcal{C} \to \mathrm{Span}(\mathrm{Set})$ factors via $\mathrm{Span}(\mathrm{FinSet})$.

The unique lifting of factorizations and finitary fiber properties characterize classical NDFAs in the following sense.

**Proposition 2.5.** *Let $\phi : \mathbb{H} \to \mathbb{G}$ be a homomorphism into a finite graph $\mathbb{G}$. Then $\mathcal{F}\ \phi : \mathcal{F}\ \mathbb{H} \to \mathcal{F}\ \mathbb{G}$ is finitary iff $\mathbb{H}$ is finite.*

*Proof.* By the infinite pigeonhole principle. □

**Proposition 2.6.** *A functor $p : \mathcal{Q} \to \mathcal{F}\ \mathbb{B}_\Sigma$ corresponds to the underlying bare automaton of a NDFA iff $p$ is ULF and finitary.*

*Proof.* By Props. 2.3 and 2.5, $p$ is ULF and finitary iff $\mathcal{Q} \cong \mathcal{F}\ \mathbb{Q}$ is generated by a finite graph with $p$ induced by a homomorphism, thereby defining the transition graph of a bare NDFA. □

This leads us to the following definition of NDFA over an arbitrary category.

**Definition 2.7.** A **non-deterministic finite state automaton over a category** is given by a tuple $M = (\mathcal{C}, \mathcal{Q}, p, q_0, q_f)$ consisting of two categories $\mathcal{C}$ and $\mathcal{Q}$, a finitary ULF functor $p : \mathcal{Q} \to \mathcal{C}$, and a pair $q_0, q_f$ of objects of $\mathcal{Q}$. An object of $\mathcal{Q}$ is then called a **state** and an arrow of $\mathcal{Q}$ is called a **run** of the automaton. The **regular language of arrows** $\mathcal{L}_M$ recognized by the automaton is the set of arrows $w$ in $\mathcal{C}$ that can be lifted along $p$ to an arrow $\alpha : q_0 \to q_f$ in $\mathcal{Q}$, that is $\mathcal{L}_M = \{\, p(\alpha) \mid \alpha : q_0 \to q_f \,\} \subseteq \mathcal{C}(A, B)$, where $p(q_0) = A, p(q_f) = B$.

Similarly to the way a categorical CFG may be considered as a functor of pointed operads $p_G : (\mathcal{F}\ \mathbb{S}, S) \to (\mathcal{W}\,\mathcal{C}, (A, B))$, it is also sometimes helpful to view a categorical NDFA as a finitary ULF functor of bipointed categories $p_M : (\mathcal{Q}, q_0, q_f) \to (\mathcal{C}, A, B)$. We emphasize that the definition does not require the base category $\mathcal{C}$ to be free, which we will see in examples later on.

Before continuing to develop this definition, let us briefly comment on our treatment of initial and accepting states, as well as the treatment of determinism and $\epsilon$-transitions. Rabin and Scott's original definition of NDFA [RS59] (also adopted by Eilenberg [Eil74]), takes a set of initial states together with a set of accepting states, while some more recent authors [HMU07, Sip13] take a single initial state together with a set of accepting states. By Rabin and Scott's determinization theorem, a language $\mathcal{L} \subseteq \Sigma^*$ may be recognized by a NDFA with a set of initial and accepting states just in case it may be recognized by a DFA with a single initial state and a set of accepting states, so the choice of having a single initial state or a set of initial states does not matter in terms of recognition power. Classically, the same is not quite true for NDFAs over a fixed alphabet with a single initial state and a single accepting state (in the absence of $\epsilon$-transitions), since the language recognized by such an automaton is necessarily closed under concatenation if it contains the empty word. Nevertheless, it is easy to see that any NDFA with a set of initial and accepting states may be converted to an equivalent NDFA with a single initial state and a single accepting state, over an alphabet extended with distinguished symbols marking the beginning and the end of the word. This construction is particularly natural to express using the category $\mathcal{F}\ \mathbb{B}_\Sigma^{\hat{}\$}$ introduced in Example 1.4.

**Proposition 2.8.** *A language $\mathcal{L} \subseteq \Sigma^*$ is regular in the classical sense if and only if $\hat{}\mathcal{L}\$$ is the language of arrows recognized by a non-deterministic finite state automaton over $\mathcal{F}\ \mathbb{B}_\Sigma^{\hat{}\$}$.*

*Proof.* Given a classical NDFA $M$ with underlying bare automaton $\mathcal{Q} \to \mathcal{F}\ \mathbb{B}_\Sigma$, we construct an NDFA $M^{\hat{}\$} = (\mathcal{F}\ \mathbb{B}_\Sigma^{\hat{}\$}, \mathcal{Q}', p' : \mathcal{Q}' \to \mathcal{F}\ \mathbb{B}_\Sigma^{\hat{}\$}, q_0, q_f)$ where $\mathcal{Q}'$ and the functor $p'$ are constructed from $\mathcal{Q}$ and $p$ by freely adjoining a pair of objects $q_0$ and $q_f$ lying over $\bot$ and $\top$ respectively, together with an arrow $q_0 \to q$ over $\hat{} : \bot \to *$ for every initial state $q$ and an arrow $q' \to q_f$ over $\$ : * \to \top$ for every accepting state $q'$ of $M$. By construction, every accepting run of $M$ over the word $w$ corresponds to an accepting run of $M^{\hat{}\$}$ over the arrow $\hat{}w\$ : \bot \to \top$. □

Although Definition 2.7 could be adapted to take a set of initial and accepting states, we find the convention of taking a single pair $(q_0, q_f)$ of initial and accepting states to align well with the definition of context-free languages of arrows generated by a single non-terminal (Definition 1.6). Let us also observe here that the category $\mathcal{F}\ \hat{\mathbb{B}}_{\Sigma}^{\$}$ in the statement of Proposition 2.8 is isomorphic to the *input category* $\mathcal{I}_{\mathsf{word}}$ used in Colcombet and Petrişan's definition of categorical automata. (See bottom of Figure 1 of [CP20]; we will give a more detailed comparison with Colcombet and Petrişan's framework in §2.5 below.)

**Definition 2.9.** An NDFA $M = (\mathcal{C}, \mathcal{Q}, p : \mathcal{Q} \to \mathcal{C}, q_0, q_f)$ is **deterministic** if the functor $p$ is a *discrete opfibration*, i.e., for any arrow $w : A \to B$ in $\mathcal{C}$ and any state $q \in \mathcal{Q}$ such that $p(q) = A$, there is a unique arrow $\alpha : q \to q'$ such that $p(\alpha) = w$. $M$ is **codeterministic** if $p$ is a *discrete fibration*, i.e., for any arrow $w : A \to B$ in $\mathcal{C}$ and any state $q' \in \mathcal{Q}$ such that $p(q') = B$, there exists a unique arrow $\alpha : q \to q'$ such that $p(\alpha) = w$. $M$ is **bideterministic** if it is both deterministic and codeterministic, i.e., $p$ is a discrete bifibration. Finally, $M$ is a **partial** (co/bi)deterministic automaton if the functor $p : \mathcal{Q} \to \mathcal{C}$ is a partial discrete op/bi/fibration, in the sense that for any $w : A \to B$ in $\mathcal{C}$ and $q \in \mathcal{Q}$ such that $p(q) = A$ (or $q'$ such that $p(q') = B$), there is at most one $\alpha : q \to q'$ such that $p(\alpha) = w$.

**Remark 2.10.** With this way of representing finite-state automata by functors $p : \mathcal{Q} \to \mathcal{C}$, $\epsilon$-transitions may be naturally interpreted as non-identity arrows $\alpha : q \to q'$ in $\mathcal{Q}$ mapped by $p$ to an identity arrow $id_A : A \to A$ in $\mathcal{C}$. However, it is an immediate consequence of the ULF property that an NDFA in the sense of Definition 2.7 cannot contain any non-trivial fiber categories $p^{-1}(A)$ (Proposition 2.2). One would therefore need a more general notion of ULF functor in order to accommodate categorical NDFAs with $\epsilon$-transitions. This seems to raise some subtle issues. In particular, the standard treatment of $\epsilon$-removal [HMU07, §2.5] seems to indicate that one needs to design a notion between ULF functors and general Conduché fibrations. Recall that the standard notion of Conduché fibration weakens the ULF property by only requiring uniqueness of factorizations up to zigzag in the fiber, which means here zigzag of $\epsilon$-transitions. The right definition of categorical NDFA with $\epsilon$-transitions should probably ensure more, that there is a universal (initial or terminal) solution in the space of possible factorizations.

As we have seen, our definition of NDFA over a category does not require the base category $\mathcal{C}$ to be free. We now list some examples of such automata and the regular languages of arrows that they generate.

**Example 2.11** (Product automata)**.** The product of two finitary ULF functors $p : \mathcal{Q} \to \mathcal{C}$ and $p' : \mathcal{Q}' \to \mathcal{C}'$ is again a finitary ULF functor $p \times p' : \mathcal{Q} \times \mathcal{Q}' \to \mathcal{C} \times \mathcal{C}'$. This enables us to construct the product of two NDFAs $M$ and $M'$ as an NDFA $M \times M'$ whose underlying bare automaton is $p \times p'$ and whose initial and accepting states are given by the pairing of the respective initial states $(q_0, q_0')$ and $(q_f, q_f')$. Note that even if $M$ and $M'$ are NDFAs over free categories $\mathcal{C} = \mathcal{F}\ \mathbb{G}$ and $\mathcal{C}' = \mathcal{F}\ \mathbb{G}'$, the product automaton $M \times M'$ will not be since the product of free categories is not a free category. Intuitively, $M \times M'$ may be interpreted as an automaton that reads pairs of arrows in parallel. It recognizes the cartesian product $\mathcal{L} \times \mathcal{L}'$ of the languages recognized by $M$ and $M'$.

**Example 2.12** (Singleton automata)**.** For any word $w = a_1 \dots a_n$ of length $n$, there is a partial bideterministic $(n+1)$-state automaton $M_w$ that recognizes the singleton language $\{ w \}$, with initial state 0, accepting state $n$, and transitions of the form $(i, a_{i+1}, i+1)$ for each $0 \leq i < n$. This construction may be generalized to define a (not necessarily deterministic

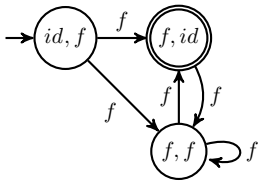
or codeterministic) NDFA $M_w$ recognizing any arrow $w$ in any category satisfying a certain finitariness assumption. Indeed, for any arrow $w : A \to B$ of a category $\mathcal{C}$, the *category of factorizations* $\mathsf{Fact}_w$ of $w$ (also called the *interval category* of $w$ in [LM10]) is defined by taking objects to be triples $(X, u, v)$ of an object $X \in \mathcal{C}$ and a pair of arrows $u : A \to X$, $v : X \to B$ such that $w = uv$, and arrows $(X, u, v) \to (X', u', v')$ to be arrows $x : X \to X'$ such that $u' = ux$ and $v = xv'$. This category has an initial object $(id_A, w)$ and a terminal object $(w, id_B)$, and it comes equipped with an evident forgetful functor $\mu_w : \mathsf{Fact}_w \to \mathcal{C}$ returning the middle of a factorization, which is always ULF [Joh99, LM10]. Hence

$$M_w = (\mathcal{C}, \mathsf{Fact}_w, \mu_w, (id_A, w), (w, id_B)) \tag{2.1}$$

may be considered as a non-deterministic automaton recognizing the singleton language $\{ w \}$, although it is not necessarily a finite-state automaton since the functor $\mu_w$ need not be finitary in general. Let us say that a category $\mathcal{C}$ **has finitary factorizations** if this is always the case, in other words, if the ULF functor $\mu_w : \mathsf{Fact}_w \to \mathcal{C}$ is finitary for every arrow $w$ of $\mathcal{C}$. Then we see that to any arrow $w$ of a category $\mathcal{C}$ with finitary factorizations, (2.1) defines an NDFA $M_w$ recognizing exactly that arrow.

Free categories always have finitary factorizations, and indeed the category of factorizations of any arrow of a free category is isomorphic to a non-empty finite ordinal, as in the example above of the $(n + 1)$-state automaton associated to a word of length $n$.

A category with finitely many objects has finitary factorizations just in case every arrow has finitely many factorizations $w = uv$ of length 2. (Note this is weaker than the condition of being a *Möbius category* in the sense of Leroux, cf. [LM10, Prop. 2.6].) For example, consider the one-object category $\mathcal{E}$ containing a single non-identity arrow $f : * \to *$ that is idempotent $ff = f$. The NDFA $M_f$ recognizing the language $\{ f \} \subseteq \mathcal{E}(*, *)$ has three states $(id, f)$, $(f, id)$, and $(f, f)$, as depicted by the following transition diagram in which all paths of length $\geq 1$ commute:




**Example 2.13** (Total automata)**.** The identity functor $id_{\mathcal{C}} : \mathcal{C} \to \mathcal{C}$ on a category is trivially ULF and finitary, since every object and arrow has a unique lifting. Hence for any pair of objects $A$ and $B$ of $\mathcal{C}$, there is an NDFA $M_{\mathcal{C}(A,B)} = (\mathcal{C}, \mathcal{C}, id_{\mathcal{C}}, A, B)$ recognizing the total language $\mathcal{C}(A, B)$, i.e., accepting every arrow $w : A \to B$. In the case of the one-object category $\mathcal{C} = \mathcal{F}\ \mathbb{B}_\Sigma$, this is just the familiar automaton with a single (initial and accepting) state and a transition on that state for every letter of $\Sigma$. On the other hand, it us worth emphasizing that $M_{\mathcal{C}(A,B)}$ defines an NDFA in our sense even when $\mathcal{C}$ has infinitely many objects. In particular, this example shows that what is important is finiteness of the *fibers* of the functor $p : \mathcal{Q} \to \mathcal{C}$ defining an NDFA, and not of the total category of states $\mathcal{Q}$.

**Example 2.14** (Asynchronous automata)**.** One good reason for considering NFDAs whose base category $\mathcal{C}$ is not free is that they provide for interesting notions of automata when the category $\mathcal{C}$ is presented by generators and equations. An *equation* in a graph $\mathbb{G}$ is a pair $f, g : A \to B$ of paths with same source and target. Every set $\mathcal{R}$ of such equations induces a congruence $\sim_{\mathcal{R}}$ on the paths of $\mathbb{G}$, defined as the smallest equivalence relation between paths of $\mathbb{G}$ with same source and target, compatible to composition, and such that $f \sim_{\mathcal{R}} g$

for every equation $f, g : A \to B$ in $\mathcal{R}$. We say that a category $\mathcal{C}$ is *presented* by a graph $\mathbb{G}$ of generators and a set $\mathcal{R}$ of equations when it comes equipped with an isomorphism $c : \mathcal{F}\ \mathbb{G}/{\sim_{\mathcal{R}}} \to \mathcal{C}$ from the category $\mathcal{F}\ \mathbb{G}/{\sim_{\mathcal{R}}}$ whose objects are nodes of $\mathbb{G}$ and whose arrows are $\sim_{\mathcal{R}}$-equivalence classes of paths.

A remarkable property of ULF functors $p : \mathcal{D} \to \mathcal{C}$ into a category $\mathcal{C}$ presented by a graph $\mathbb{G}$ and a set $\mathcal{R}$ of equations, with isomorphism $c : \mathcal{F}\ \mathbb{G}/{\sim_{\mathcal{R}}} \to \mathcal{C}$, is that the category $\mathcal{D}$ inherits a presentation by a graph $p^*\,\mathbb{G}$ and a set $p^*\,\mathcal{R}$ of equations derived from $\mathbb{G}$ and $\mathcal{R}$. The graph $p^*\,\mathbb{G}$ has nodes the pairs $(x, A)$ of a node $x$ in $\mathbb{G}$ and of an object $A$ of $\mathcal{D}$ such that $c(x) = p(A)$, and edges the pairs $(e, u)$ of an edge $e : x \to y$ in $\mathbb{G}$ and an arrow $u : A \to B$ in $\mathcal{D}$ such that $c(e) = p(u)$. The set $p^*\,\mathcal{R}$ contains all equations between paths of $p^*\,\mathbb{G}$ of the form

$$f = (e_1, u_1) \cdots (e_m, u_m)\,,\ g = (e'_1, v_1) \cdots (e'_n, v_n) \quad : \quad (x, A) \longrightarrow (y, B)$$

such that

$$e_1 \cdots e_m\,,\ e'_1 \cdots e'_n \quad : \quad x \longrightarrow y$$

is an equation in $\mathcal{R}$ and the composite arrows

$$u_1 \cdots u_m\,,\ v_1 \cdots v_n \quad : \quad A \longrightarrow B$$

are equal in the category $\mathcal{D}$. The isomorphism $d : p^*\,\mathbb{G}/{\sim_{p^*\,\mathcal{R}}} \to \mathcal{D}$ is given by the second projection $(x, A) \mapsto A$ on objects, and by the map $[e_1 \cdots e_m, u_1 \cdots u_m] \mapsto u_1 \cdots u_m$ sending an equivalence class of lists of edge/arrow pairs to the composition of the arrows in $\mathcal{D}$.

We illustrate the property by giving a direct description of NFDAs on a category $\mathcal{C}$ defined by a Mazurkiewicz alphabet. A concurrent alphabet $(\Sigma, D)$ as defined by Mazurkiewicz [Maz89] is a set $\Sigma$ of letters, equipped with a reflexive and symmetric binary relation $D \subseteq \Sigma \times \Sigma$. Two letters $a, b \in \Sigma$ are called *dependent* when $a, b \in D$ and *independent* otherwise. The *permutation equivalence* on finite words is defined as the smallest equivalence relation $\sim$ such that $w_1\, a\, b\, w_2 \sim w_1\, b\, a\, w_2$ for every pair of words $w_1, w_2 \in \Sigma^*$ and independent letters $a, b \in \Sigma$. A *trace* is an equivalence class of the permutation equivalence. Traces $w \in \Sigma^*/{\sim}$ may be regarded as the arrows $w : * \to *$ of a one-object category $\mathcal{F}\ \mathbb{B}_\Sigma/{\sim_D}$ obtained by quotienting the category $\mathcal{F}\ \mathbb{B}_\Sigma$ with an equation between arrows $a\, b = b\, a : * \to *$ for every pair of independent letters $a, b \in \Sigma$.

By unpacking the definitions, a NFDA over $\mathcal{F}\ \mathbb{B}_\Sigma/{\sim_D}$ may be equivalently described as follows, starting from the data of a finite set $Q$ of states, a finite set *Tran* of transitions together with a labelling function $\delta : \mathit{Tran} \to Q \times \Sigma \times Q$, and a choice of initial and accepting states. The transition graph is moreover equipped with a set $\diamond$ of permutation tiles defined as pairs $(\alpha, \beta)$ of paths $\alpha = st$ and $\beta = t's'$ of length 2 of the form

$$\delta(s) = (q, a, q'),\ \delta(t) = (q', b, q''),\ \delta(t') = (q, b, q''') \text{ and } \delta(s') = (q''', a, q'')$$

where $q, q', q'', q''' \in Q$ are states and the two letters $a, b \in \Sigma$ are independent. We write $\alpha \diamond \beta$ when $(\alpha, \beta) \in \diamond$. The set of permutation tiles is required to satisfy the following properties:

(1) permutation tiles are deterministic: $\alpha \diamond \beta$ and $\alpha \diamond \beta'$ implies $\beta = \beta'$,
(2) permutation tiles are symmetric: $\alpha \diamond \beta$ implies $\beta \diamond \alpha$,
(3) successive transitions with independent letters can permute: for every transition path $\alpha = s\,t$ of length 2 with $\delta(s) = (q, a, q')$ and $\delta(t) = (q', b, q'')$ where the two letters $a, b \in \Sigma$ are independent, there exists a permutation tile $\alpha \diamond \beta$.

The correspondence works as follows. Given a finite transition system $(Q, \mathit{Tran}, \delta)$ equipped with a set $\diamond$ of permutation tiles, one defines the category $\mathcal{C}$ presented by the corresponding graph $\mathbb{G}$ with an equation $\alpha = \beta$ for every permutation tile $\alpha \diamond \beta$ ; the three properties (1), (2), and (3) ensure then that the labelling functor $q : \mathcal{C} \to \mathcal{F}\ \mathbb{B}_\Sigma/\!\sim_D$ is finitary ULF. Conversely, every finitary ULF functor $p : \mathcal{C} \to \mathcal{F}\ \mathbb{B}_\Sigma/\!\sim_D$ induces a finite graph $\mathbb{G} = p^*\,\mathbb{B}_\Sigma$ and a set of equations $\diamond = p^*\,\mathcal{R}_D$ where $\mathcal{R}_D$ denotes the set of equations $ab, ba : * \to *$ for $a$ and $b$ two independent letters ; one checks that the nodes and edges of $\mathbb{G}$ define a transition system $(Q, \mathit{Tran}, \delta)$ which satisfies properties (1), (2), and (3). Note that property (3) reflects the ULF property for the factorisation as $b$ followed by $a$ of the arrow $ab$ in $\mathcal{F}\ \mathbb{B}_\Sigma/\!\sim_D$, for any pair of independent letters $a$ and $b$ in $\Sigma$.

This establishes that the notion of NFDA over the category $\mathcal{F}\ \mathbb{B}_\Sigma/\!\sim_D$ of Mazurkiewicz traces captures a natural notion of asynchronous automaton, corresponding to a non-deterministic version of Bednarczyk's asynchronous systems [Bed88, HM00, Mor05, BM06] where property (3) is called the Independent Diamond (ID) property. A detailed comparison of this class of asynchronous automaton with the more liberal notion of automaton with concurrency relations [DS02, GM10] appears in [Mor05]. The ID property implies that every NFDA over $\mathcal{F}\ \mathbb{B}_\Sigma/D$ satisfies the cube property (CP) discussed in [HM00] and thus defines what is called an asynchronous graph in [MM07, Mel21]. We leave for future work the connection with Zielonka's theorem which is based on a more concrete and process-based notion of concurrent automaton [Zie87, BM06].

2.2. **From word automata to tree automata and beyond.** Reassuringly, the foregoing fibrational analysis of non-deterministic finite state automata based on finitary ULF functors adapts smoothly when one shifts from word automata to tree automata [CDG$^+$08]. As a first step in that direction, we describe how the ULF and finite fiber properties may be extended to functors of operads.

**Definition 2.15.** A functor of operads $p : \mathcal{D} \to \mathcal{O}$ has the **unique lifting of factorizations property** (or **is ULF**) if any of the following equivalent conditions hold:

(1) for any operation $\alpha$ of $\mathcal{D}$, if $p(\alpha) = g \circ (h_1, \dots, h_n)$ for some operation $g$ and list of operations $h_1, \dots, h_n$ of $\mathcal{O}$, there exists a unique operation $\beta$ and list of operations $\gamma_1, \dots, \gamma_n$ of $\mathcal{D}$ such that $\alpha = \beta \circ (\gamma_1, \dots, \gamma_n)$ and $p(\beta) = g$, $p(\gamma_1) = h_1, \dots, p(\gamma_n) = h_n$;
(2) for any operation $\alpha$ of $\mathcal{D}$, if $p(\alpha) = g \circ_i h$ for some operations $g$ and $h$ of $\mathcal{O}$ and index $i$, there exists a unique pair of operations $\beta$ and $\gamma$ of $\mathcal{D}$ such that $\alpha = \beta \circ_i \gamma$ and $p(\beta) = g$ and $p(\gamma) = h$;
(3) the structure maps of the associated lax functor of operads $F : \mathcal{O} \to \mathrm{Span}(\mathrm{Set})$ discussed in §1.5 are invertible.

**Definition 2.16.** We say that a functor of operads $p : \mathcal{D} \to \mathcal{O}$ is **finitary** if either of the following equivalent conditions hold:

- the fiber $p^{-1}(A)$ as well as the fiber $p^{-1}(f)$ is finite for every color $A$ and operation $f$ of the operad $\mathcal{O}$;
- the associated lax functor of operads $F : \mathcal{O} \to \mathrm{Span}(\mathrm{Set})$ factors via $\mathrm{Span}(\mathrm{FinSet})$.

Just as we saw in the previous section, one can easily check that the underlying bare automaton $M = (\Sigma, Q, \delta)$ of a (bottom-up or top-down) non-deterministic finite state tree automaton (cf. §1.1 and §1.6 of [CDG$^+$08]) gives rise to a finitary ULF functor of operads $p : \mathcal{Q} \to \mathcal{F}\ \Sigma$, where

- $\mathcal{F}\ \Sigma$ is the free operad generated by the ranked alphabet (= species) $\Sigma$;
- the operad $\mathcal{Q}$ has states of the automaton as colors and operations freely generated by $n$-ary nodes of the form $t : q_1, \ldots, q_n \to q$ for every transition $t \in \mathit{Tran}$ of the form $\delta(t) = (q_1, \ldots, q_n, a, q)$, with $a \in \Sigma_n$;
- the functor $p$ transports every such $n$-ary transition $t : q_1, \ldots, q_n \to q$ to the underlying $n$-ary letter $a : *, \ldots, * \to *$.

Moreover, this correspondence is one-to-one.

**Proposition 2.17.** *Let $p : \mathcal{D} \to \mathcal{O}$ be a functor into an operad $\mathcal{O} = \mathcal{F}\ \Sigma$ freely generated by some species $\Sigma$. Then $p$ is ULF iff $\mathcal{D} = \mathcal{F}\ \mathbb{S}$ is free over some species $\mathbb{S}$ and $p = \mathcal{F}\ \phi$ is generated by a map of species $\phi : \mathbb{S} \to \Sigma$.*

**Proposition 2.18.** *A functor of operads $p : \mathcal{Q} \to \mathcal{F}\ \Sigma$ corresponds to a bare non-deterministic finite state tree automaton iff $p$ is ULF and finitary.*

This motivates us to proceed as for word automata and propose a more general notion of finite state automaton over an arbitrary operad.

**Definition 2.19.** A **non-deterministic finite state automaton over an operad** is given by a tuple $M = (\mathcal{O}, \mathcal{Q}, p : \mathcal{Q} \to \mathcal{O}, q_r)$ consisting of two operads $\mathcal{O}$ and $\mathcal{Q}$, a finitary ULF functor of operads $p : \mathcal{Q} \to \mathcal{O}$, and a color $q_r$ of $\mathcal{Q}$. A color of $\mathcal{Q}$ is called a **state**, and an operation of $\mathcal{Q}$ is called a **run tree** of the automaton $p : \mathcal{Q} \to \mathcal{O}$. The **regular language of constants** $\mathcal{L}_M$ recognized by the automaton is the set of constants $c$ in $\mathcal{O}$ that can be lifted along $p$ to a constant $\alpha : q_r$ in $\mathcal{Q}$, that is $\mathcal{L}_M = \{\, p(\alpha) \mid \alpha : q_r \,\} \subseteq \mathcal{O}(A)$, where $p(q_r) = A$.

Let us refer to NDFAs over categories in the sense of Definition 2.7 as *categorical automata* for short, and NDFAs over operads in the foregoing sense as *operadic automata*.

We now state a simple property of ULF and finitary functors establishing a useful connection between categorical automata and operadic automata.

**Proposition 2.20.** *Suppose that $p : \mathcal{Q} \to \mathcal{C}$ is a functor of categories. If $p$ is ULF (respectively, finitary) then so is the functor of operads $\mathcal{W}\,p : \mathcal{W}\,\mathcal{Q} \to \mathcal{W}\,\mathcal{C}$.*

*Proof.* Follows immediately from the fact that a lifting of an operation $f_0 - \ldots - f_n$ in $\mathcal{W}\,\mathcal{C}$ along $\mathcal{W}\,p$ is a sequence of liftings of the arrows $f_0, \ldots, f_n$ in $\mathcal{C}$ along $p$. □

**Corollary 2.21.** *For any categorical automaton $M = (\mathcal{C}, \mathcal{Q}, p, q_0, q_f)$ there is an associated operadic automaton $\mathcal{W}\,M = (\mathcal{W}\,\mathcal{C}, \mathcal{W}\,\mathcal{Q}, \mathcal{W}\,p, (q_0, q_f))$ generating the same language $\mathcal{L}_M = \mathcal{L}_{\mathcal{W}\,M}$.*

*Proof.* By Proposition 2.20, and because constants of $\mathcal{W}\,\mathcal{C}$ are exactly arrows of $\mathcal{C}$. □

We will see that the $\mathcal{W}\,M$ construction transforming an automaton over a category into an automaton over its operad of spliced arrows is fundamental for computing the intersection of a context-free language with a regular language of arrows. Moreover, we highlight that since operads of spliced arrows are not freely generated (Remark 1.3), the $\mathcal{W}\,M$ construction provides an important example of how operadic automata truly generalize classical tree automata.

**Remark 2.22.** As an aside, we mention that it is also interesting to consider *deterministic* finite-state automata over spliced arrow operads. To see why, let us first observe that any discrete opfibration over $\mathcal{W}\,\mathcal{C}$, which may be viewed equivalently as a functor of operads

$\mathcal{W}\mathcal{C} \to \mathrm{Set}$, induces a category $\mathcal{D}$ equipped with an identity-on-objects functor of categories $\mathcal{C} \to \mathcal{D}$, and conversely. Indeed, given a functor $F : \mathcal{W}\mathcal{C} \to \mathrm{Set}$ one can construct a new category $\mathcal{D}_F$ with same objects as $\mathcal{C}$ and whose arrows $\alpha : A \to B$ are given by the elements of $F(A, B)$, with composition and identities defined by the interpretations

$$F(id_A - id_B - id_C) : F(A, B) \times F(B, C) \longrightarrow F(A, C)$$
$$F(id_A) : F(A, A)$$

of the corresponding spliced arrow operations. Moreover, there is an identity-on-objects functor $\mathcal{C} \to \mathcal{D}_F$ given by sending every arrow $w : A \to B$ of $\mathcal{C}$, seen as a constant $w : (A, B)$ of $\mathcal{W}\mathcal{C}$, to its interpretation $F(w) : F(A, B)$. Conversely, given a category $\mathcal{D}$ and an identity-on-objects functor $h : \mathcal{C} \to \mathcal{D}$, one can define $F_h : \mathcal{W}\mathcal{C} \to \mathrm{Set}$ by taking $F_h(A, B) = \mathcal{D}(A, B)$ and defining the interpretation of the spliced arrow operations by mapping the arrows of $\mathcal{C}$ into $\mathcal{D}$ along $h$ and then composing:

$$F_h(w_0 - w_1 \ldots - w_n) = (u_1, \ldots, u_n) \mapsto h(w_0)u_1h(w_1) \ldots u_nh(w_n)$$

From the description of this correspondence, it is clear it restricts to one between *finitary* discrete opfibrations over $\mathcal{W}\mathcal{C}$ (equivalent to functors $\mathcal{W}\mathcal{C} \to \mathrm{FinSet}$) and *locally finite* categories $\mathcal{D}$ equipped with an identity-on-objects functor $\mathcal{C} \to \mathcal{D}$. Thus what is a DFA over $\mathcal{W}\mathcal{C}$? Nothing more than a locally finite category $\mathcal{D}$, an identity-on-objects functor $h : \mathcal{C} \to \mathcal{D}$, and an arrow $q_r : A \to B$ of $\mathcal{D}$, with the DFA accepting an arrow $w : A \to B$ of $\mathcal{C}$ just in case $h(w) = q_r$. In the case that $\mathcal{C}$ is a one-object category, this is essentially the notion of *recognition by a finite monoid,* which is one of the classical equivalent characterizations of regular languages [Eil74, §III.10, p.62].[5]

2.3. **Additional properties of ULF and finitary functors.** We state here a few more basic properties of ULF and finitary functors, in both categorical and operadic versions, that will be important in the rest of the paper.

**Proposition 2.23.** *If $p : Y \to X$ is a ULF (resp. finitary) functor (of categories or operads), and $G : Z \to X$ is an arbitrary functor, then the functor $G^* p : Z \times_X Y \to Z$ defined by pulling back $p$ along $G$ is ULF (resp. finitary).*

$$\begin{array}{ccc} Z \times_X Y & \longrightarrow & Y \\ {\scriptstyle G^*p \ \text{ULF/finitary}}\downarrow & \lrcorner & \downarrow{\scriptstyle p \ \text{ULF/finitary}} \\ Z & \xrightarrow[G]{} & X \end{array}$$

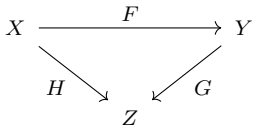
Closure of both classes of functors under pullback along arbitrary functors may be verified directly, or may be seen as an immediate consequence of the fact that both classes of functors $p : Y \to X$ admit "Grothendieck-type" correspondences with pseudo or lax functors $F : X \to \mathcal{U}$ into a certain universe (namely, ULF functors as pseudo functors into $\mathcal{U} = \mathrm{Span}(\mathrm{Set})$, and finitary functors as lax functors into $\mathcal{U} = \mathrm{Span}(\mathrm{FinSet})$), so that the pullback along an ordinary functor $G : Z \to X$ is simply the composite $F \circ G : Z \to \mathcal{U}$.

As we already saw in Example 2.13, the identity functor $id_{\mathcal{C}} : \mathcal{C} \to \mathcal{C}$ on any category is trivially finitary and ULF, as is the identity functor $id_{\mathcal{O}} : \mathcal{O} \to \mathcal{O}$ on any operad. It is also

[5] A language $\mathcal{L} \subseteq \Sigma^*$ is said to be recognizable by a finite monoid $X$ if there exists a subset $S \subseteq X$ and a homomorphism $h : \Sigma^* \to X$ such that $\mathcal{L} = h^{-1}(S)$. Our formulation requires $S$ to be a singleton, cf. Proposition 2.8 above and the preceding discussion.

easy to check that both classes of functors are closed under composition. Moreover, ULF functors satisfy a property analogous to the pasting law for pullbacks in a category, while finitary functors behave similarly to monomorphisms.

**Proposition 2.24.** *Consider a commutative triangle of functors (of categories or operads):*



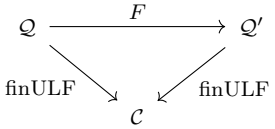

(1) *if $F$ and $G$ are ULF, then so is $H$*
(2) *if $F$ and $G$ are finitary, then so is $H$*
(3) *if $G$ and $H$ are ULF then so is $F$*
(4) *if $H$ is finitary then so is $F$*

*Proof.* For concreteness, we take $X$, $Y$, and $Z$ to be categories. (The case where they are operads is completely analogous.)

(1) Let $\alpha$ be an arrow in $X$. Any factorization $H(\alpha) = uv$ in $Z$ lifts uniquely to a factorization of $\alpha = \beta\gamma$ with $H(\beta) = u$, $H(\gamma) = v$, by first lifting along $G$ to obtain a unique factorization of $F(\alpha)$, and then lifting along $F$.
(2) All of the fibers of $H$ are finite (for both the objects and arrows of $Z$) since they may be decomposed as a finite union of finite fibers $H^{-1}(z) = \bigcup_{y \in G^{-1}(z)} F^{-1}(y)$.
(3) Let $\alpha$ be an arrow in $X$ and $F(\alpha) = st$ a factorization in $Y$. By projecting down along $G$ and then lifting the resulting factorization $H(\alpha) = G(s)G(t)$ along $H$, we obtain unique $\beta$ and $\gamma$ such that $\alpha = \beta\gamma$, $H(\beta) = G(s)$, $H(\gamma) = G(t)$. But since $F(\alpha) = F(\beta)F(\gamma)$ is a lifting of the factorization $G(st) = G(s)G(t)$ along $G$, this implies that $F(\beta) = s$ and $F(\gamma) = t$ by the ULF property of $G$.
(4) All of the fibers $F^{-1}(y)$ are finite, since they are contained in the fibers $H^{-1}(G(y))$.

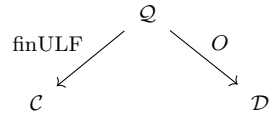
□

We remark in passing that finitary ULF functors may be arranged into various categories and bicategories of interest, which we describe now for the case of categorical automata (the operadic case again being completely analogous), although we will not make explicit use of these (bi)categories in the paper. For any category $\mathcal{C}$, finitary ULF functors over $\mathcal{C}$ and commutative triangles

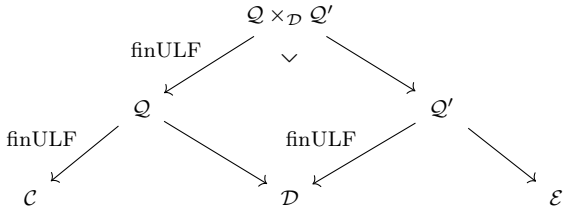



define a category of bare NDFAs over $\mathcal{C}$ and morphisms between them, where the functor $F : \mathcal{Q} \to \mathcal{Q}'$ may be seen as a **functional simulation** since it sends any run $\alpha : q_0 \to q_f$ in $\mathcal{Q}$ over an arrow $w : A \to B$ of $\mathcal{C}$ to a run $F(\alpha) : F(q_0) \to F(q_f)$ in $\mathcal{Q}'$ over the same arrow. By Proposition 2.24, $F$ is itself finitary ULF, and this category is just the slice finULF/$\mathcal{C}$ of the category finULF of small categories and finitary ULF functors between them. It is possible to formulate analogues of the weak and the strong translation principles (Proposition 1.11): if $M$ and $M'$ are two NDFAs over the same category, and $F : \mathcal{Q} \to \mathcal{Q}'$ is a functional simulation between their underlying bare automata that preserves initial and accepting states, then $\mathcal{L}_M \subseteq \mathcal{L}_{M'}$, where equality holds $\mathcal{L}_M = \mathcal{L}_{M'}$ in the case that $F$ is full and faithful (i.e., if it is a *functional bisimulation*).

In a different direction of abstraction, a span




of a finitary ULF functor $\mathcal{Q} \to \mathcal{C}$ and an arbitrary functor $O : \mathcal{Q} \to \mathcal{D}$ may be considered as a **non-deterministic finite state transducer** $\mathcal{C} \nrightarrow \mathcal{D}$, since it associates to every run $\alpha : q_0 \to q_f$ in $\mathcal{Q}$ over an arrow $w : A \to B$ of $\mathcal{C}$ a corresponding output arrow $O(\alpha) : O(q_0) \to O(q_f)$ in $\mathcal{D}$. By Propositions 2.23 and 2.24, ordinary composition of spans




combines a transducer $\mathcal{C} \nrightarrow \mathcal{D}$ with a transducer $\mathcal{D} \nrightarrow \mathcal{E}$ to produce a transducer $\mathcal{C} \nrightarrow \mathcal{E}$, and one obtains a bicategory by defining a 2-cell between transducers of the same type as a functional simulation between the underlying bare automata that commutes with the output functors.

2.4. **Closure properties of regular languages.** Closure of finitary ULF functors under pullback along arbitrary functors (Proposition 2.23) implies that regular languages of arrows and constants are closed under inverse functorial image, restricted to a suitable choice of objects in the source category/operad.

**Proposition 2.25.** *Let $\mathcal{L} \subseteq \mathcal{C}(A, B)$ be a regular language of arrows in $\mathcal{C}$, let $F : \mathcal{D} \to \mathcal{C}$ be a functor of categories, and let $R, S$ be objects of $\mathcal{D}$ such that $F(R) = A, F(S) = B$. Then the homset-restricted inverse image $F^{-1}(\mathcal{L}) \cap \mathcal{D}(R, S)$ is regular.*

**Proposition 2.26.** *Let $\mathcal{L} \subseteq \mathcal{O}(A)$ be a regular language of constants in $\mathcal{O}$, let $F : \mathcal{P} \to \mathcal{O}$ be a functor of operads, and let $R$ be a color of $\mathcal{P}$ such that $F(R) = A$. Then the conset-restricted inverse image $F^{-1}(\mathcal{L}) \cap \mathcal{P}(R)$ is regular.*

Note that restricting the inverse image to a homset/conset is necessary due to our conventions for the initial and accepting states of categorical and operadic automata (although variations of those conventions could certainly be envisaged, see discussion above after Def. 2.7). Indeed, given a categorical automaton $M = (\mathcal{C}, \mathcal{Q}, p, q_0, q_f)$, a functor $F : \mathcal{D} \to \mathcal{C}$, and a pair of objects $R, S$ of $\mathcal{D}$ such that $F(R) = p(q_0), F(S) = p(q_f)$, the restricted pullback automaton is given by $F^* M|_{R,S} = (\mathcal{D}, \mathcal{D} \times_{\mathcal{C}} \mathcal{Q}, F^* p, (R, q_0), (S, q_f))$. The restricted pullback of an operadic automaton is constructed similarly.

Conversely, since finitary ULF functors are closed under composition (Proposition 2.24), we immediately obtain closure of regular languages under functorial image along a finitary ULF functor.

**Proposition 2.27.** *If $\mathcal{L} \subseteq \mathcal{C}(A, B)$ is a regular language of arrows in $\mathcal{C}$ and $F : \mathcal{C} \to \mathcal{D}$ is a finitary ULF functor of categories, then the image $F(\mathcal{L}) \subseteq \mathcal{D}(F(A), F(B))$ is also regular.*

**Proposition 2.28.** *If $\mathcal{L} \subseteq \mathcal{O}(A)$ is a regular language of constants in $\mathcal{O}$ and $F : \mathcal{O} \to \mathcal{P}$ is a finitary ULF functor of operads, then the image $F(\mathcal{L}) \subseteq \mathcal{P}(F(A))$ is also regular.*

For example, the pushforward of a categorical automaton $M = (\mathcal{C}, \mathcal{Q}, p, q_0, q_f)$ along a functor $F : \mathcal{C} \to \mathcal{D}$ is the automaton $F_* M = (\mathcal{D}, \mathcal{Q}, F \circ p, q_0, q_f)$.

On the other hand, regular languages are *not* closed under image along an arbitrary functor, as the following counterexamples demonstrate.

**Counterexample 2.29.** Let $\mathcal{W}$ be the *category of Dyck walks* freely generated by the graph whose nodes are natural numbers and with a pair of edges

$$n \underset{d_n}{\overset{u_n}{\rightleftarrows}} n+1$$

for every $n \in \mathbb{N}$. Let $\Sigma$ be the alphabet containing a pair of brackets '[' and ']' and consider the functor $F : \mathcal{W} \to \mathcal{F}\ \mathbb{B}_\Sigma$ defined by

$$n \mapsto * \quad u_n \mapsto [ \quad d_n \mapsto ]$$

for every $n \in \mathbb{N}$. The total language $\mathcal{W}(0,0) \subseteq \mathcal{W}(0,0)$ is regular (Example 2.13), but its image $F(\mathcal{W}(0,0)) \subseteq \mathcal{F}\ \mathbb{B}_\Sigma(*,*)$ is the Dyck language of well-bracketed words, which is not a regular language in the classical sense [HMU07, p. 195] and hence cannot be a regular language of arrows in our sense by Prop. 2.6. Observe in this case that the functor $F$ is ULF since it is generated from a graph homomorphism, but it is manifestly not finitary.

**Counterexample 2.30.** Let $2 = \ 0 \xrightarrow{f} 1$ be the walking arrow category, and consider any functor $F : 2 \to \mathcal{C}$, which picks out an arrow $F(f) = w : A \to B$ in $\mathcal{C}$. Observe that the functor $F$ is finitary, but will typically not be ULF unless $w$ is indecomposable. The singleton language $\{ f \}$ is regular (Example 2.12), and its image $F(\{ f \}) = \{ w \} \subseteq \mathcal{C}(A, B)$ will be as well if the category $\mathcal{C}$ has finitary factorizations, but otherwise need not be regular. In particular, let $\mathcal{C} = \mathcal{B}_{\mathbb{R}_{\geq 0}}$ be the monoid of non-negative real numbers $(\mathbb{R}_{\geq 0}, +, 0)$ considered as a one-object category, and let $w = 1$ be the unit value considered as an arrow $1 : * \to *$. We claim that the language $\{ 1 \} \subseteq \mathbb{R}_{\geq 0}$ is not regular. Suppose by way of contradiction that there is an automaton $M$ recognizing $\{ 1 \}$ equipped with a finitary ULF functor $p : \mathcal{Q} \to \mathcal{B}_{\mathbb{R}_{\geq 0}}$ and initial and accepting states $q_0$ and $q_f$. Since 1 factors in the monoid as

$$1 = \frac{1}{n} + \frac{n-1}{n}$$

for all $n \geq 1$, the accepting run $\alpha : q_0 \to q_f$ of 1 factors as

$$\alpha = \ q_0 \xrightarrow{\beta_n} q_n \xrightarrow{\gamma_n} q_f$$

for some unique arrows $\beta_n$ and $\gamma_n$ lying over $\frac{1}{n}$ and $\frac{n-1}{n}$, by assumption that $p$ is ULF. Now there are two possibilities:

(1) The objects $q_n$ are distinct for all $n \geq 1$. In this case the fiber $p^{-1}(*)$ is infinite, contradicting the assumption that $p$ is finitary.
(2) There exist $i < j$ such that $q_i = q_j$. In this case

$$\alpha' = \ q_0 \xrightarrow{\alpha_i} q_i = q_j \xrightarrow{\gamma_j} q_f$$

is another accepting run of the automaton over the element $p(\alpha') = \frac{1}{i} + \frac{j-1}{j} > 1$, contradicting the assumption that $M$ recognizes just the singleton language $\{ 1 \}$.

Finally, we can easily derive that regular languages are closed under intersection.

**Proposition 2.31.** *If $\mathcal{L}_1, \mathcal{L}_2 \subseteq \mathcal{C}(A, B)$ (resp. $\mathcal{L}_1, \mathcal{L}_2 \subseteq \mathcal{O}(A)$) are regular languages of arrows (resp. constants), so is their intersection $\mathcal{L}_1 \cap \mathcal{L}_2$.*

*Proof.* Given two automata $M_1$ and $M_2$ with underlying bare automata $p_1 : \mathcal{Q}_1 \to \mathcal{C}$ and $p_2 : \mathcal{Q}_2 \to \mathcal{C}$ over the same category $\mathcal{C}$, and with initial and accepting states mapped to the same pair of objects $A$ and $B$, the intersection automaton $M_1 \cap M_2$ can be constructed in either of two equivalent ways:

(1) By pulling back $p_1$ along $p_2$, and then taking the image along $p_2$.
(2) By forming the product automaton $M_1 \times M_2$ (Example 2.11) over $\mathcal{C} \times \mathcal{C}$ and then pulling back along the diagonal functor $\mathcal{C} \to \mathcal{C} \times \mathcal{C}$.

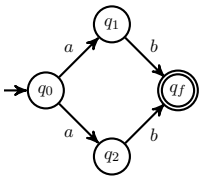
The construction of the intersection of operadic automata is completely analogous. □

At this point, we should mention that we expect regular languages of arrows and constants to satisfy some additional closure properties just as in the classical case, including suitably formulated versions of closure under union, concatenation, and Kleene star. However, the usual proof of these facts is greatly facilitated by the use of $\epsilon$-transitions, which are ruled out by our working definition of categorical and operadic NDFAs based on ULF functors (see Remark 2.10). We therefore defer consideration of these additional closure properties to the future introduction of an appropriate notion of categorical/operadic automaton with $\epsilon$-transitions. In any case, we will not need them to prove the representation theorem.

2.5. **Related work.** It is standard to represent finite state automata by their transition graphs, and natural to consider such transition graphs as ordinary (finite directed) graphs equipped with a homomorphism into the bouquet graph on the underlying alphabet. Such a perspective on non-deterministic finite state automata without $\epsilon$-transitions was developed by Steinberg [Ste01], motivated by monoid and semigroup theory, and with special prominence given to graph homomorphisms that are *immersions,* which correspond to partial bideterministic automata in the sense of Definition 2.9. Walters' short note on context-free languages [Wal89] that we mentioned in §1.9 also opens with a treatment of automata as graph homomorphisms, but taking morphisms of *reflexive graphs* in order to allow $\epsilon$-transitions. As discussed in Remark 2.10, an important question is how to adapt our approach based on finitary ULF functors to allow for $\epsilon$-transitions, although we are confident that this can be done by appropriately generalizing the notion of ULF. Walters' approach to automata and context-free grammars was further developed by Rosenthal [Ros95], considering also tree automata, and more recently by Earnshaw and Sobociński [ES22] to define regular languages in free monoidal categories.

The fibrational perspective on non-deterministic finite state automata over categories and operads that we introduced here is closely related to the "functorial approach" to automata theory developed by Colcombet and Petrişan [CP20]. Roughly speaking, the approaches are dual and related by Grothendieck-type correspondences: whereas we consider automata as functors into a category (or operad) with certain fibrational properties, Colcombet and Petrişan consider them as functors *out of* a category into different large categories of interest, such as Set for deterministic automata, Rel for non-deterministic automata, or Vec for weighted automata. Given that in many cases such Grothendieck-type correspondences rise to the level of equivalences, generally the approaches provide two complementary perspectives on the same objects. Still, it is worth drawing attention to a few small but

important differences between our approach and the approach described in [CP20]. First, although many of the definitions and constructions given by Colcombet and Petrişan are stated for an arbitrary "input category" $\mathcal{I}$, in practice they consider automata over a fixed category $\mathcal{I}_{\mathsf{word}}$, isomorphic to the category we call $\mathcal{F}\ \hat{\mathbb{B}}_{\Sigma}^{\$}$, and they do not impose any finiteness condition on automata. Second, rather than having a single initial and accepting state, Colcombet and Petrişan take a more abstract definition of the "behavior" of an automaton given by precomposing with a full and faithful functor $\iota : \mathcal{O} \to \mathcal{I}$ from some "output category" $\mathcal{O}$, typically the full subcategory of $\mathcal{I}_{\mathsf{word}} \cong \mathcal{F}\ \hat{\mathbb{B}}_{\Sigma}^{\$}$ spanned by the arrows $\bot \to \top$. The idea is that precomposing with $\iota$ abstracts away the internal states of the automaton, remembering just the language that it recognizes. Colcombet and Petrişan use this to give an elegant analysis of automata minimization based on factorization systems.




Finally, and although it is by no means forced by the functorial approach, let us observe that modelling non-deterministic automata as functors into Rel (rather than as pseudofunctors into Span(Set) or Span(FinSet)) means equating all of the runs over a given word between a given pair of states. For instance, the NDFA shown on the left has two distinct runs $q_0 \to q_f$ over the word $ab$, but these are identified when considering it as a Rel-automaton.

The idea of generalizing regular languages from recognizable subsets of free monoids $\mathcal{L} \subseteq \Sigma^*$ to recognizable subsets of arbitrary monoids $\mathcal{L} \subseteq X$ is classic. For instance, it is discussed in Eilenberg's book (see [Eil74, §III.12, p.68]) using several of the equivalent characterizations of regular languages, including recognition by a finite monoid. He mentions that the definition of automaton could also be suitably generalized, but does not do so explicitly "since all the known properties of recognizable sets already follow from the definition adopted above". There is also precedent for the idea of generalizing from languages of words in free monoids to languages of arrows in free categories, having been proposed at least as early as the mid-1980s in the work of Thérien and collaborators [TSG88, WT86], who argued that this was well-motivated by automata-theoretic considerations:

> For example, we are often interested in decompositions of automata. In such situations a component may receive its input from the output of some other component. This "preprocessing" imposes restrictions on the possible input sequences that need to be considered. A simple way to take into account these restrictions is to view a machine as processing input sequences that are paths in a finite directed multigraph. [TSG88, p.395]

Again, though, rather than taking NDFAs or DFAs as a starting point, these authors started from the alternative characterization of regular word languages as unions of equivalence classes of a finite index congruence on a free monoid, and explained how to generalize it to free categories in a way that allowed proving Kleene's theorem among other results. We take one more opportunity to emphasize that our definition of regular language is not restricted to free categories and operads, and that non-free categories and operads are required in some key examples. (The work by Thérien et al. was, incidentally, revisited by Jones who took a similar approach to define regular languages in a certain class of *profinite categories* [Jon96].) We do *not* expect Kleene's theorem on the equivalence between regular and rational languages to hold for regular languages of arrows in arbitrary categories – indeed classically it is a theorem about recognizable subsets of free monoids with counterexamples in the non-free case [Eil74, §IV.5, p.175–178], and Counterexample 2.30 demonstrates that some

singleton languages of arrows cannot be recognized by any NDFA. On the other hand, we mention that Fahrenberg et al. have recently proved a Kleene theorem for higher-dimensional automata recognizing languages of interval pomsets [FJSZ22], which it would be interesting to analyze within our framework.

Although it is beyond the scope of this paper, we expect that regular languages of arrows/constants in the sense of Definitions 2.7 and 2.19 should also have other equivalent characterizations related to the aforementioned classical ones – in particular there seem to be natural notions of *recognition by a locally finite category* and *recognition by a locally finite operad* (cf. Remark 2.22). It may be fruitful to connect this with Salvati's notion of regular language of $\lambda$-terms [Sal09] (in part inspired by [dG01]), which has recently been formulated as recognition by a locally finite and well-pointed cartesian closed category [MN24].

Finally, we only briefly touched upon the subject of constructing categories of automata and transducers in §2.3, but this also has precedent in the literature. Our definition of a non-deterministic finite state transducer as a span $\mathcal{C} \twoheadleftarrow \mathcal{Q} \to \mathcal{D}$ where the left leg satisfies a fibrational condition (namely, is finitary and ULF) is similar to the notion of "Mealy morphism" defined by Paré using slightly different conditions [Par12].[6] Hyland defined a *traced monoidal category* Aut whose objects are natural numbers and whose morphisms $m \to n$ are finite-state automata (over a fixed alphabet $\Sigma$) equipped with $m$ distinct initial states and $n$ distinct accepting states [Hyl08]. He then exhibited a traced monoidal functor $\mathrm{Aut} \to \mathrm{Rat}$ to a category whose arrows are matrices of rational languages, as one direction of Kleene's theorem. It could be interesting to adapt Hyland's analysis to our setting.

## 3. The Chomsky-Schützenberger Representation Theorem

In this section, we give a relatively simple and conceptual proof of the representation theorem, generalized to context-free languages of arrows in an arbitrary category. We begin in §3.1 by explaining how to represent the intersection of a context-free language with a regular language by first taking the *pullback* of a CFG along an NDFA. Next, in §3.2, we exhibit a left adjoint to the spliced arrow operad construction, which we call the *contour category* construction. As a direct consequence of this adjunction, we show in §3.3 that every pointed finite species induces a CFG that is universal in a precise sense and that generates a language of *tree contour words,* which are closely related to Dyck words. Finally, in §3.4 we state and prove an appropriate generalization of the representation theorem, which also relies in a crucial way on the strong translation principle.

### 3.1. **Pulling back context-free grammars along finite state automata.**

To compute the pullback, we apply two lemmas.

**Lemma 3.1.** *Suppose given a species $\mathbb{S}$, a functor of operads $p : \mathcal{F}\,\mathbb{S} \to \mathcal{O}$ and a ULF functor of operads $p_{\mathcal{Q}} : \mathcal{Q} \to \mathcal{O}$. In that case, the pullback of $p$ along $p_{\mathcal{Q}}$ in the category of operads is obtained from a corresponding pullback of $\phi : \mathbb{S} \to \mathcal{O}$ along $p_{\mathcal{Q}} : \mathcal{Q} \to \mathcal{O}$ in the*

[6] Although in our context, a transducer $\mathcal{C} \twoheadleftarrow \mathcal{Q} \to \mathcal{D}$ is better seen as a (non-deterministic) *Moore machine,* since the output in $\mathcal{D}$ only depends on $\mathcal{Q}$, i.e., on the automaton's internal states and transitions.

*category of species:*

$$\begin{array}{ccc} \mathbb{S}' & \xrightarrow{\psi'} & \mathbb{S} \\ {\scriptstyle \phi' = p_{\mathcal{Q}}^* \phi}\downarrow & \lrcorner & \downarrow{\scriptstyle \phi} \\ \mathcal{Q} & \xrightarrow[p_{\mathcal{Q}}]{} & \mathcal{O} \end{array} \quad \in \quad \text{Spec} \qquad \Bigg\| \qquad \begin{array}{ccc} \mathcal{F}\,\mathbb{S}' & \xrightarrow{\mathcal{F}\,\psi'} & \mathcal{F}\,\mathbb{S} \\ {\scriptstyle p' = p_{\mathcal{Q}}^* p}\downarrow & \lrcorner & \downarrow{\scriptstyle p} \\ \mathcal{Q} & \xrightarrow[p_{\mathcal{Q}}]{} & \mathcal{O} \end{array} \quad \in \quad \text{Oper}$$

*Proof.* By Proposition 2.23, the pullback of the ULF functor $p_{\mathcal{Q}}$ along the functor of operads $p$ is a ULF functor $\mathcal{Q}' \to \mathcal{F}\,\mathbb{S}$, which must be of the form $\mathcal{F}\,\psi' : \mathcal{F}\,\mathbb{S}' \to \mathcal{F}\,\mathbb{S}$ for some map of species $\psi' : \mathbb{S}' \to \mathbb{S}$ by Proposition 2.17. We therefore have a pullback in Oper as on the right above, which is sent to a pullback in Spec by the right adjoint forgetful functor. The unit of the adjunction induces another pullback diagram

$$\begin{array}{ccc} \mathbb{S}' & \xrightarrow{\psi'} & \mathbb{S} \\ {\scriptstyle \eta_{\mathbb{S}}}\downarrow & \lrcorner & \downarrow{\scriptstyle \eta_{\mathbb{S}}} \\ \mathcal{F}\,\mathbb{S}' & \xrightarrow{\mathcal{F}\,\psi'} & \mathcal{F}\,\mathbb{S} \end{array}$$

and by composing these

$$\begin{array}{ccc} \mathbb{S}' & \xrightarrow{\psi'} & \mathbb{S} \\ {\scriptstyle \eta_{\mathbb{S}}}\downarrow & \lrcorner & \downarrow{\scriptstyle \eta_{\mathbb{S}}} \\ \mathcal{F}\,\mathbb{S}' & \xrightarrow{\mathcal{F}\,\psi'} & \mathcal{F}\,\mathbb{S} \\ {\scriptstyle p'}\downarrow & \lrcorner & \downarrow{\scriptstyle p} \\ \mathcal{Q} & \xrightarrow[p_{\mathcal{Q}}]{} & \mathcal{O} \end{array} \quad = \quad \begin{array}{ccc} \mathbb{S}' & \xrightarrow{\psi'} & \mathbb{S} \\ {\scriptstyle \phi'}\downarrow & \lrcorner & \downarrow{\scriptstyle \phi} \\ \mathcal{Q} & \xrightarrow[p_{\mathcal{Q}}]{} & \mathcal{O} \end{array}$$

we obtain the desired pullback in the category of species. □

**Lemma 3.2.** *Consider a pullback in the category of species:* $\begin{array}{ccc} \mathbb{S}' & \xrightarrow{\psi'} & \mathbb{S} \\ {\scriptstyle \phi'}\downarrow & \lrcorner & \downarrow{\scriptstyle \phi} \\ \mathbb{R}' & \xrightarrow[\psi]{} & \mathbb{R} \end{array}$*. If* $\mathbb{S}$ *is finite and* $\psi$ *is finitary in the expected sense (i.e., has finite fibers) then* $\mathbb{S}'$ *is finite.*

*Proof.* The pullback species $\mathbb{S}' = \mathbb{S} \times_{\mathbb{R}} \mathbb{R}'$ admits a simple description: its colors (resp. nodes) are pairs of colors (resp. nodes) from $\mathbb{S}$ and $\mathbb{R}'$ lying over the same color (resp. node) in $\mathbb{R}$. Hence finiteness of $\mathbb{S}'$ follows immediately from the assumption that $\mathbb{S}$ is finite and that $\psi$ has finite fibers. □

**Theorem 3.3.** *For any CFG* $G = (\mathcal{C}, \mathbb{S}, S, p_G)$ *and NDFA* $M = (\mathcal{C}, \mathcal{Q}, p_M, q_0, q_f)$ *over the same category, with* $p_G(S) = (p_M(q_0), p_M(q_f))$*, there is a CFG* $G' = M^* G$ *generating the language* $\mathcal{L}_{G'} = p_M^{-1}(\mathcal{L}_G) \cap \mathcal{Q}(q_0, q_f)$*.*

*Proof.* By Prop. 2.20, the finitary ULF functor of categories $p_M : \mathcal{Q} \to \mathcal{C}$ induces a finitary ULF functor of operads $\mathcal{W}\,p_M : \mathcal{W}\,\mathcal{Q} \to \mathcal{W}\,\mathcal{C}$. By Lemma 3.1, the pullback of $p_G$ along $p_M$

may therefore be computed as a pullback of the underlying maps of species:

$$\begin{array}{ccc} \mathbb{S}' & \xrightarrow{\psi'} & \mathbb{S} \\ {\scriptstyle \phi'_G}\downarrow & \lrcorner & \downarrow{\scriptstyle \phi_G} \\ \mathcal{W}\,\mathcal{Q} & \xrightarrow[\mathcal{W}\,p_M]{} & \mathcal{W}\,\mathcal{C} \end{array} \quad \in \quad \mathrm{Spec} \qquad \Big\| \qquad \begin{array}{ccc} \mathcal{F}\,\mathbb{S}' & \xrightarrow{\mathcal{F}\,\psi'} & \mathcal{F}\,\mathbb{S} \\ {\scriptstyle p'_G}\downarrow & \lrcorner & \downarrow{\scriptstyle p_G} \\ \mathcal{W}\,\mathcal{Q} & \xrightarrow[\mathcal{W}\,p_M]{} & \mathcal{W}\,\mathcal{C} \end{array} \quad \in \quad \mathrm{Oper}$$

Moreover, by Lemma 3.2, the pullback species $\mathbb{S}' = \mathbb{S} \times_{\mathcal{W}\,\mathcal{C}} \mathcal{W}\,\mathcal{Q}$ is finite, so that $M^*\,G = (\mathcal{Q}, \mathbb{S}', (S, (q_0, q_f)), p'_G)$ defines a CFG. Finally, we have $\mathcal{L}_{G'} = p_M^{-1}(\mathcal{L}_G) \cap \mathcal{Q}(q_0, q_f)$ because by the universal property of the pullback in Oper, $G'$ derives a run $\alpha : q_0 \to q_f$ of $M$ just in case there exists a constant $\beta : S$ in $\mathcal{F}\,\mathbb{S}$ such that $p_G(\beta) = p_M(\alpha)$. □

We emphasize that the pullback grammar $M^*\,G$ generates a language of arrows in $\mathcal{Q}$, corresponding to the runs of the automaton $M$ over the arrows in the language generated by the original grammar $G$. The pullback grammar admits the following concrete description in traditional CFG syntax:

- its non-terminals are pairs $(R, (q, q'))$ – which for geometric intuition we prefer to visualize as triples $(q, R, q')$ – for every non-terminal $R$ of the grammar $G$ and pair of states $q, q'$ of the automaton $M$ such that $p_G(R) = (p_M(q), p_M(q'))$;
- it has a production rule $(q, R, q') \to \alpha_0(q_1, R_1, q'_1)\alpha_1 \dots (q_n, R_n, q'_n)\alpha_n$ for every production rule $R \to w_0 R_1 w_1 \dots R_n w_n$ of $G$ and sequence of $n+1$ runs $\alpha_0 : q \to q_1$, $\alpha_1 : q'_1 \to q_2$, …, $\alpha_n : q'_n \to q'$ of $M$ over the respective arrows $w_0, \dots, w_n$;
- the start symbol is $(q_0, S, q_f)$, where $S$ is the start symbol of $G$ and $q_0$ and $q_f$ are the initial and accepting states of $M$.

**Example 3.4.** Recall from Example 2.1 that to any arrow $w : A \to B$ of a category with finitary factorizations $\mathcal{C}$, there is associated an NDFA $M_w$ recognizing exactly the singleton language $\{\, w \,\}$. If $G$ is a CFG over $\mathcal{C}$, then by pulling back $G$ along $M_w$ we obtain a new grammar $M_w^*\,G$ that may be seen as a specialization of $G$ to the factors of $w$. In particular, in the classical case where $w = a_1 \dots a_n$ is a word of length $n$ over $\Sigma$, the non-terminals of $M_w^*\,G$ may be taken as triples $(i, R, j)$ of a non-terminal $R$ of $G$ together with a pair of indices $0 \leq i, j \leq n$, generating all the parses of the subword $w_{i,j} = a_{i+1} \dots a_j$ as an $R$. If we then view this grammar as defining a displayed operad over $\mathcal{W}\,\mathsf{Fact}_w$, as discussed in §1.5–§1.6, we obtain precisely the classical parse matrices $N_{i,j}$ referenced in §1.7.

As an immediate corollary of Theorem 3.3, we obtain the closure of CFLs of arrows under intersection with regular languages of arrows.

**Corollary 3.5.** *If $\mathcal{L} \subseteq \mathcal{C}(A, B)$ is context-free and $\mathcal{L}_M \subseteq \mathcal{C}(A, B)$ is regular then $\mathcal{L} \cap \mathcal{L}_M \subseteq \mathcal{C}(A, B)$ is context-free.*

*Proof.* By taking the functorial image (Prop. 1.9(3)) of the CFL $p_M^{-1}(\mathcal{L}_G) \cap \mathcal{Q}(q_0, q_f)$ along the functor $p_M$. □

Explicitly, the grammar generating $L \cap L_M$ is defined by starting with the pullback grammar $G' = M^*\,G$ from Theorem 3.3 above and postcomposing the functor $p'_G : \mathcal{F}\,\mathbb{S}' \to \mathcal{W}\,\mathcal{Q}$ with $\mathcal{W}\,p_M : \mathcal{W}\,\mathcal{Q} \to \mathcal{W}\,\mathcal{C}$. This grammar admits almost the same concrete description as $G'$ but where the production rules instead take the form

$$(q, R, q') \to w_0(q_1, R_1, q'_1)w_1 \dots (q_n, R_n, q'_n)w_n$$

for every production rule $R \to w_0 R_1 w_1 \dots R_n w_n$ of $G$ and sequence of $n+1$ runs $q \to q_1$, $q'_1 \to q_2$, …, $q'_n \to q'$ of $M$ over the respective arrows $w_0, \dots, w_n$. The reader familiar with the classic "Bar-Hillel construction" for combining a CFG with an NDFA [BHPS61, Theorem 8.1] may recognize that our construction is very similar, but completely uniform and with the benefit of being derived systematically.

Although it will not be relevant to the representation theorem for categorical CFGs, let us observe that the construction of the pullback of a CFG along a categorical NDFA works just as well to construct the pullback of a generalized CFG (in the sense of §1.8) along an operadic NDFA.

**Theorem 3.6.** *For any gCFG $G = (\mathcal{O}, \mathbb{S}, S, p_G)$ and operadic NDFA $M = (\mathcal{O}, \mathcal{Q}, p_M, q_r)$ over the same operad, with $p_G(S) = p_M(q_r)$, there is a gCFG $G' = M^* G$ generating the language $\mathcal{L}_{G'} = p_M^{-1}(\mathcal{L}_G) \cap \mathcal{Q}(q_r)$.*

Indeed, the proof is almost identical to the proof of Theorem 3.3, simply omitting the first step where we applied the functor $\mathcal{W}$ before applying Lemma 3.1. We immediately obtain that gCFLs are closed under intersection with regular languages of constants.

**Corollary 3.7.** *If $\mathcal{L} \subseteq \mathcal{O}(A)$ is a gCFL and $\mathcal{L}_M \subseteq \mathcal{O}(A)$ is regular then $L \cap \mathcal{L}_M \subseteq \mathcal{O}(A)$ is a gCFL.*

3.2. **The contour category of an operad and the contour / splicing adjunction.** In §1.2, we began by explaining how to construct a functor $\mathcal{W} : \mathrm{Cat} \to \mathrm{Oper}$ transforming any category $\mathcal{C}$ into an operad $\mathcal{W}\mathcal{C}$ of spliced arrows of arbitrary arity, which played a central role in our definition of context-free language of arrows in a category. We construct now a left adjoint functor

$$\mathrm{Oper} \underset{\mathcal{W}}{\overset{\mathcal{C}}{\underset{\longleftarrow}{\overset{\longrightarrow}{\bot}}}} \mathrm{Cat} \tag{3.1}$$

which extracts from any given operad $\mathcal{O}$ a category $\mathcal{C}\,\mathcal{O}$ whose arrows may be interpreted as "oriented contours" along the boundary of the operations of the operad.

**Definition 3.8.** The **contour category** $\mathcal{C}\,\mathcal{O}$ of an operad $\mathcal{O}$ is defined as a quotient of the following free category:

- objects are given by *oriented colors* $R^\epsilon$ consisting of a color $R$ of $\mathcal{O}$ and an orientation $\epsilon \in \{ u, d \}$ ("up" or "down");
- arrows are generated by pairs $(f, i)$ of an operation $f : R_1, \dots, R_n \to R$ of $\mathcal{O}$ and an index $0 \leq i \leq n$, defining an arrow $R_i^d \to R_{i+1}^u$ under the conventions that $R_0^d = R^u$ and $R_{n+1}^u = R^d$;

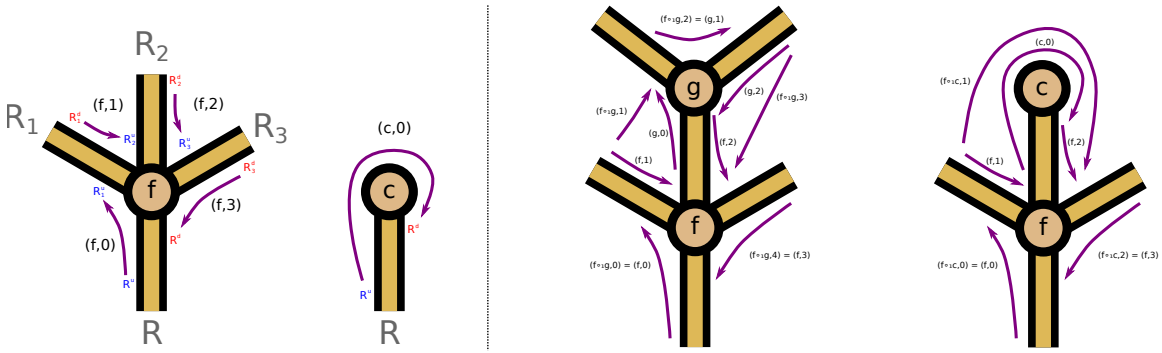



Figure 5: Left: interpretation of the generating arrows of the contour category $\mathcal{C}\,\mathcal{O}$. Right: interpretation of equations (3.2) and (3.3).

subject to the conditions that $id_{R^u} = (id_R, 0)$ and $id_{R^d} = (id_R, 1)$ as well as the following equations:

$$(f \circ_i g, j) = \begin{cases} (f,j) & j < i \\ (f,i)(g,0) & j = i \\ (g, j-i) & i < j < i+m \\ (g,m)(f,i+1) & j = i+m \\ (f, j-m+1) & j > i+m \end{cases} \tag{3.2}$$

$$(f \circ_i c, j) = \begin{cases} (f,j) & j < i \\ (f,i)(c,0)(f,i+1) & j = i \\ (f,j+1) & j > i \end{cases} \tag{3.3}$$

whenever the left-hand side is well-formed, for every operation $f$, operation $g$ of positive arity $m > 0$, constant $c$, and indices $i$ and $j$ in the appropriate range.

We refer to each generating arrow $(f,i)$ of the contour category $\mathcal{C}\,\mathcal{O}$ as a **sector** of the operation $f$. See Fig. 5 for a graphical interpretation of sectors and of the equations on contours seen as compositions of sectors.

The contour construction provides a left adjoint to the spliced arrow construction because a functor of operads $\mathcal{O} \to \mathcal{W}\,\mathcal{C}$ is entirely described by the data of a pair of objects $(A,B) = (R^u, R^d)$ in $\mathcal{C}$ for every color $R$ in $\mathcal{O}$ together with a sequence $f_0, f_1, \ldots, f_n$ of $n+1$ arrows in $\mathcal{C}$, where $f_i : R_i^d \to R_{i+1}^u$ for $0 \leq i \leq n$ for each operation $f : R_1, \ldots, R_n \to R$ of $\mathcal{O}$, under the same conventions as above. The equations (3.2) and (3.3) on the generators of $\mathcal{C}\,\mathcal{O}$ reflect the equations imposed by the functor of operads $\mathcal{O} \to \mathcal{W}\,\mathcal{C}$ on the spliced arrows of $\mathcal{C}$ appearing as the image of operations in $\mathcal{O}$. In that way we transform any functor of operads $\mathcal{O} \to \mathcal{W}\,\mathcal{C}$ into a functor $\mathcal{C}\,\mathcal{O} \to \mathcal{C}$ which may be seen as an interpretion of the contours of the operations of $\mathcal{O}$ in $\mathcal{C}$.

The unit and counit of the contour / splicing adjunction also have nice descriptions. The unit of the adjunction defines, for any operad $\mathcal{O}$, a functor of operads $\mathcal{O} \to \mathcal{W}\,\mathcal{C}\,\mathcal{O}$ that acts on colors by $R \mapsto (R^u, R^d)$, and on operations by sending an operation $f : R_1, \ldots, R_n \to R$ of $\mathcal{O}$ to the spliced word of sectors $(f,0) - \ldots - (f,n) : (R_1^u, R_1^d), \ldots, (R_n^u, R_n^d) \to (R^u, R^d)$. The counit of the adjunction defines, for any category $\mathcal{C}$, a functor of categories $\mathcal{C}\,\mathcal{W}\,\mathcal{C} \to \mathcal{C}$

that acts on objects by $(A,B)^u \mapsto A$ and $(A,B)^d \mapsto B$, and on arrows by sending the $i$th sector of a spliced word to its $i$th word, $(w_0 - \ldots - w_n, i) \mapsto w_i$.

**Remark 3.9.** In the case of a free operad over a species $\mathbb{S}$, the contour category $\mathcal{C}\,\mathcal{F}\,\mathbb{S}$ admits an even simpler description as a free category $\mathcal{C}\,\mathcal{F}\,\mathbb{S} \cong \mathcal{F}\,\mathcal{C}\,\mathbb{S}$ generated by the arrows $(x,i) : R_i^d \to R_{i+1}^u$ for every node $x : R_1, \ldots, R_n \to R$ of the species $\mathbb{S}$. Indeed, it is easy to verify that there is an analogous adjunction between species and graphs, and that this pair of contour / splicing adjunctions commutes with the pair of respective free / forgetful adjunctions:

$$\begin{array}{ccc} & \xrightarrow{\quad\mathcal{C}\quad} & \\ \mathrm{Oper} & \perp & \mathrm{Cat} \\ \uparrow \dashv \downarrow & \xleftarrow{\quad\mathcal{W}\quad} & \uparrow \dashv \downarrow \\ & \xrightarrow{\quad\mathcal{C}\quad} & \\ \mathrm{Spec} & \perp & \mathrm{Grph} \\ & \xleftarrow{\quad\mathcal{W}\quad} & \end{array} \tag{3.4}$$

We refer to the generating arrows $(x,i)$ of the contour graph $\mathcal{C}\,\mathbb{S}$ as **corners** in the sense of the theory of planar maps [Sch15], since geometrically they indeed correspond to the corners of $\mathbb{S}$-rooted trees seen as rooted planar maps. By (3.4), every sector of an operation of $\mathcal{F}\,\mathbb{S}$ factors uniquely in the contour category $\mathcal{C}\,\mathcal{F}\,\mathbb{S}$ as a sequence of corners.

In contrast to the situation for $\mathcal{W}$ (Prop. 2.20), it is *not* the case that $\mathcal{C}$ always preserves the ULF property.

**Remark 3.10.** Consider the category $\mathbf{1}+\mathbf{1}$ with two objects $A$ and $B$ and only identity arrows, and the unique functor $p$ to the terminal category $\mathbf{1}$. We claim that the associated ULF functor of operads $\mathcal{W}\,p$ induces a functor of categories $\mathcal{C}\,\mathcal{W}\,p$ which is not ULF. Consider the two binary operations $f = id_A - id_A - id_A$ and $g = id_A - id_A - id_B$ and the constant $c = id_A$ in $\mathcal{W}\,(\mathbf{1}+\mathbf{1})$, as well as the binary operations $h = id_* - id_* - id_*$ and the constant $d = id_*$ in $\mathcal{W}\,\mathbf{1}$. The category $\mathcal{C}\,\mathcal{W}\,(\mathbf{1}+\mathbf{1})$ has the sequence of sectors $\alpha = (f,0)(c,0)(g,1)$ as an arrow, which is different from the identity. On the other hand, it is mapped by $\mathcal{C}\,\mathcal{W}\,p$ to the sequence $w = (h,0)(d,0)(h,1)$, which is equal thanks to Equation (3.3) to the sector $(h \circ_0 d, 0)$ of the unary operation $h \circ_0 d = id_* - id_*$ of $\mathcal{W}\,\mathbf{1}$, and hence $w = id_{(*,*)^u}$. Since the factorization $id = idid$ in $\mathcal{W}\,\mathbf{1}$ lifts to two distinct factorizations $\alpha = id\alpha = \alpha id$ in $\mathcal{W}\,(\mathbf{1}+\mathbf{1})$, $p$ is not ULF.

Still, we can verify that maps of species induce ULF functors between their contour categories, as an immediate consequence of Prop. 2.3 and the factorization (3.4).

**Proposition 3.11.** *If $\psi : \mathbb{S} \to \mathbb{R}$ is a map of species, then $\mathcal{C}\,\mathcal{F}\,\psi : \mathcal{C}\,\mathcal{F}\,\mathbb{S} \to \mathcal{C}\,\mathcal{F}\,\mathbb{R}$ is a ULF functor of categories.*

3.3. **The universal context-free grammar of a pointed species, and its associated tree contour language.** Every finite species $\mathbb{S}$ equipped with a color $S$ comes with a *universal* context-free grammar $\mathsf{Univ}_{\mathbb{S},S} = (\mathcal{C}\,\mathcal{F}\,\mathbb{S}, \mathbb{S}, S, p_{\mathbb{S}})$, characterized by the fact that $p_{\mathbb{S}} : \mathcal{F}\,\mathbb{S} \to \mathcal{W}\,\mathcal{C}\,\mathcal{F}\,\mathbb{S}$ is the unit of the contour / splicing adjunction. By "universal" context-free grammar, we mean that any categorical context-free grammar $G = (\mathcal{C}, \mathbb{S}, S, p)$ with the same underlying species and start symbol factors uniquely through $\mathsf{Univ}_{\mathbb{S},S}$ in the sense that there exists a unique functor $q_G : \mathcal{C}\,\mathcal{F}\,\mathbb{S} \to \mathcal{C}$ satisfying the equation

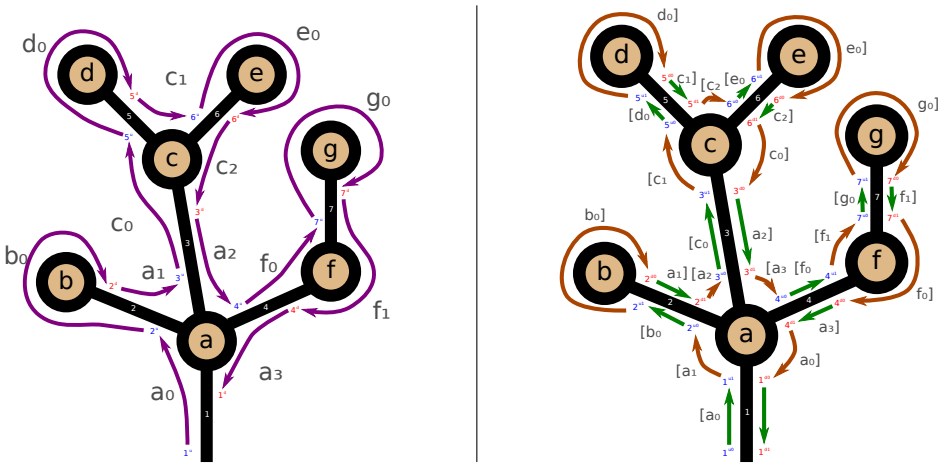



Figure 6: Left: an $\mathbb{S}$-rooted tree of root color 1 and its corresponding contour word $\mathsf{a}_0\mathsf{b}_0\mathsf{a}_1\mathsf{c}_0\mathsf{d}_0\mathsf{c}_1\mathsf{e}_0\mathsf{c}_2\mathsf{a}_2\mathsf{f}_0\mathsf{g}_0\mathsf{f}_1\mathsf{a}_3 : 1^u \to 1^d$. Right: the corresponding Dyck word obtained by first decomposing each corner of the contour into alternating actions of walking along an edge and turning around a node.

$$\mathcal{F}\,\mathbb{S} \xrightarrow{\;p\;} \mathcal{W}\,\mathcal{C} \quad = \quad \mathcal{F}\,\mathbb{S} \xrightarrow{\;p_{\mathbb{S}}\;} \mathcal{W}\,\mathcal{C}\,\mathcal{F}\,\mathbb{S} \xrightarrow{\;\mathcal{W}\,q_G\;} \mathcal{W}\,\mathcal{C}$$

We refer to the language of arrows $TC_{\mathbb{S},S} := \mathcal{L}_{\mathsf{Univ}_{\mathbb{S},S}}$ as a **tree contour language**, and to its arrows as **tree contour words**, since they describe the contours of $\mathbb{S}$-rooted trees with root color $S$, see left side of Fig. 6 for an illustration. The factorization above shows that any context-free grammar $G$ is the functorial image of the universal grammar $\mathsf{Univ}_{\mathbb{S},S}$ along the functor of categories $q_G$, whose purpose is to transport each corner of a node in $\mathbb{S}$ to the corresponding arrow in $\mathcal{C}$ as determined by the grammar. At the level of languages, we have

**Proposition 3.12.** $\mathcal{L}_G = q_G\, TC_{\mathbb{S},S}$.

**Example 3.13.** Consider the context-free grammar of Figure 2. The production rules of the universal grammar $\mathsf{Univ}_{\mathbb{S},S}$ may be depicted in traditional CFG syntax as

$$\begin{aligned} \mathsf{S} &\to 1_0\,\mathsf{NP}\,1_1\,\mathsf{VP}\,1_2 \\ \mathsf{NP} &\to 2_0 \\ \mathsf{NP} &\to 3_0 \\ \mathsf{VP} &\to 4_0\,\mathsf{NP}\,4_1 \end{aligned}$$

where we have written $i_j$ as shorthand for the sector $(x_i, j)$. The language $\mathcal{L}_G$ is then obtained as the image of $TC_{\mathbb{S},S}$ under the functor $q_G : \mathcal{C}\,\mathcal{F}\,\mathbb{S} \to \mathcal{F}\,\mathbb{B}_\Sigma$ defined by:

$$1_0 \mapsto id \quad 1_1 \mapsto \sqcup \quad 1_2 \mapsto id$$

$$2_0 \mapsto \mathsf{mom} \qquad 3_0 \mapsto \mathsf{tom} \qquad 4_0 \mapsto \mathsf{loves}_{\sqcup} \quad 4_1 \mapsto id$$

**Remark 3.14.** The notion of tree contour language makes sense even for infinite pointed species $(\mathbb{S}, S)$, although in that case the resulting universal grammar $\mathsf{Univ}_{\mathbb{S},S}$ is no longer a CFG, having infinitely many non-terminals. Still, it may be an interesting object of study. In particular, the tree contour language $\mathsf{Univ}_{\mathbb{N},*}$ generated by the *terminal species* $\mathbb{N}$ with one color and a single operation of every arity appears to be of combinatorial interest, with

words in the language describing the shapes of rooted planar trees with arbitrary node degrees. Nevertheless, from now on when we refer to a "tree contour language" we will always mean the context-free tree contour language generated by a finite pointed species.

3.4. **Representation theorem.** The achievement of the classical Chomsky-Schützenberger representation theorem is to separate any context-free grammar $G = (\Sigma, N, S, P)$ into two independent components:

(1) a context-free grammar $G'$ with only one non-terminal over an alphabet
$$\Sigma_{2n} = \{\, [_1, ]_1, \ldots, [_n, ]_n \,\}$$
of size $2n$, for some $n$, which generates Dyck words of balanced brackets describing the shapes of parse trees with nodes labelled by production rules of $G$; and
(2) a finite state automaton $M$ to check that the edges of these trees may be appropriately colored by the non-terminals of $G$ according to the labels of the nodes specifying the productions.

The original context-free language generated by $G$ is then obtained as the image of the intersection of the Dyck language generated by $G'$ with the regular language recognized by $M$, under a homomorphism $\Sigma_{2n}^* \to \Sigma^*$ that interprets each bracket of the Dyck word by a word in the original alphabet, with an arbitrary choice to either interpret the open or the close brackets as empty words.

In this section, we give a new proof of the representation theorem, generalized to context-free grammars $G$ over any category $\mathcal{C}$. Since the category $\mathcal{C}$ may have more than one object, the appropriate statement of the representation theorem cannot require the grammar describing the shapes of parse trees to have only one non-terminal. Nonetheless, we can construct one that is $\mathcal{C}$*-chromatic* in the following sense.

**Definition 3.15.** A categorical context-free grammar $G = (\mathcal{C}, \mathbb{S}, S, p)$ is **$\mathcal{C}$-chromatic** if the functor $p : \mathcal{F}\,\mathbb{S} \to \mathcal{W}\mathcal{C}$ is injective on colors.

The non-terminals of a $\mathcal{C}$-chromatic grammar may thus be considered as pairs $(A, B)$ of objects of $\mathcal{C}$.

Moreover, rather than using Dyck words to represent parse trees, we find it more natural to use tree contour words, based on the observation given above (Prop. 3.12) that every context-free language may be *canonically* represented as the image of a tree contour language generated by a context-free grammar with the same set of non-terminals. (We will discuss the relationship between contour words and Dyck words as Remark 3.20 below.)

As preparation to our proof of the representation theorem, we establish:

**Proposition 3.16.** *Any map of species $\phi : \mathbb{S} \to \mathbb{R}$ factors as*
$$\mathbb{S} \xrightarrow{\ \phi\ } \mathbb{R} \quad = \quad \mathbb{S} \xrightarrow{\ \phi_{\mathsf{colors}}\ } \phi_C\,\mathbb{S} \xrightarrow{\ \phi_{\mathsf{nodes}}\ } \mathbb{R}$$
*for some species $\phi_C\,\mathbb{S}$ such that $\phi_{\mathsf{colors}}$ is the identity on nodes and surjective on colors and $\phi_{\mathsf{nodes}}$ is injective on colors.*

*Proof.* A map of species is given by a commutative diagram as on the right below:
$$\begin{array}{ccccccc}
\mathbb{S} & & C_{\mathbb{S}}^* & \xleftarrow{\ i\ } & V_{\mathbb{S}} & \xrightarrow{\ o\ } & C_{\mathbb{S}} \\
\downarrow{\scriptstyle \phi} & & \downarrow{\scriptstyle \phi_C^*} & & \downarrow{\scriptstyle \phi_V} & & \downarrow{\scriptstyle \phi_C} \\
\mathbb{R} & & C_{\mathbb{R}}^* & \xleftarrow{\ i'\ } & V_{\mathbb{R}} & \xrightarrow{\ o'\ } & C_{\mathbb{R}}
\end{array}$$

After taking the epi-mono factorization of $\phi_C$,

$$C_{\mathbb{S}} \xrightarrow{\phi_C} C_{\mathbb{R}} \;=\; C_{\mathbb{S}} \xrightarrow{e} \!\!\!\rightarrow\; \phi_C\, C_{\mathbb{S}} \rightarrowtail^{m} C_{\mathbb{R}}$$

we obtain the desired factorization of $\phi$:

$$\begin{array}{ccccccc}
\mathbb{S} & & C^*_{\mathbb{S}} & \xleftarrow{\;i\;} & V_{\mathbb{S}} & \xrightarrow{\;o\;} & C_{\mathbb{S}} \\
\downarrow{\scriptstyle \phi_{\mathsf{colors}}} & & \downarrow{\scriptstyle e^*} & & \| & & \downarrow{\scriptstyle e} \\
\phi_C\,\mathbb{S} & & (\phi_C\, C_{\mathbb{S}})^* & \xleftarrow{\;ie^*\;} & V_{\mathbb{S}} & \xrightarrow{\;oe\;} & \phi_C\, C_{\mathbb{S}} \\
\downarrow{\scriptstyle \phi_{\mathsf{nodes}}} & & \downarrow{\scriptstyle m^*} & & \downarrow{\scriptstyle \phi_V} & & \downarrow{\scriptstyle m} \\
\mathbb{R} & & C^*_{\mathbb{R}} & \xleftarrow{\;i'\;} & V_{\mathbb{R}} & \xrightarrow{\;o'\;} & C_{\mathbb{R}}
\end{array}$$

□

**Lemma 3.17.** *Every map of species $\psi : \mathbb{S} \to \mathbb{S}'$ injective on nodes induces a commutative diagram*

$$\begin{array}{ccc}
\mathcal{F}\,\mathbb{S} & \xrightarrow{\mathcal{F}\,\psi} & \mathcal{F}\,\mathbb{S}' \\
{\scriptstyle p_{\mathbb{S}}}\downarrow & & \downarrow{\scriptstyle p_{\mathbb{S}'}} \\
\mathcal{WCF}\,\mathbb{S} & \xrightarrow[\mathcal{WCF}\,\psi]{} & \mathcal{WCF}\,\mathbb{S}'
\end{array} \tag{3.5}$$

*where the canonical functor of operads from $\mathcal{F}\,\mathbb{S}$ to the pullback of $p_{\mathbb{S}'}$ along $\mathcal{WCF}\,\psi$ is fully faithful.*

*Proof.* Every map of species $\psi : \mathbb{S} \to \mathbb{S}'$ induces a pair of naturality squares

$$\begin{array}{ccc}
\mathbb{S} & \xrightarrow{\psi} & \mathbb{S}' \\
{\scriptstyle \psi_{\mathbb{S}}}\downarrow & & \downarrow{\scriptstyle \phi_{\mathbb{S}'}} \\
\mathcal{WCF}\,\mathbb{S} & \xrightarrow[\mathcal{WCF}\,\psi]{} & \mathcal{WCF}\,\mathbb{S}'
\end{array}
\qquad\Big\|\qquad
\begin{array}{ccc}
\mathcal{F}\,\mathbb{S} & \xrightarrow{\mathcal{F}\,\psi} & \mathcal{F}\,\mathbb{S}' \\
{\scriptstyle p_{\mathbb{S}}}\downarrow & & \downarrow{\scriptstyle p_{\mathbb{S}'}} \\
\mathcal{WCF}\,\mathbb{S} & \xrightarrow[\mathcal{WCF}\,\psi]{} & \mathcal{WCF}\,\mathbb{S}'
\end{array} \tag{3.6}$$

in Spec and Oper respectively, where the functors of operads $p_{\mathbb{S}}$ and $p_{\mathbb{S}'}$ associated to the universal grammars are the units of the contour / splicing adjunction. Note that these naturality squares will in general *not* be pullback squares. By Lemma 3.1, we know that the pullback of $p_{\mathbb{S}'}$ along $\mathcal{WCF}\,\psi$ is obtained from a corresponding pullback in the category of species

$$\begin{array}{ccc}
\mathbb{R} & \xrightarrow{\pi} & \mathbb{S}' \\
{\scriptstyle \rho}\downarrow & \lrcorner & \downarrow{\scriptstyle \phi_{\mathbb{S}'}} \\
\mathcal{WCF}\,\mathbb{S} & \xrightarrow[\mathcal{WCF}\,\psi]{} & \mathcal{WCF}\,\mathbb{S}'
\end{array}
\qquad\Big\|\qquad
\begin{array}{ccc}
\mathcal{F}\,\mathbb{R} & \xrightarrow{\mathcal{F}\,\pi} & \mathcal{F}\,\mathbb{S} \\
{\scriptstyle r}\downarrow & \lrcorner & \downarrow{\scriptstyle p_{\mathbb{S}'}} \\
\mathcal{WCF}\,\mathbb{S} & \xrightarrow[\mathcal{WCF}\,\psi]{} & \mathcal{WCF}\,\mathbb{S}'
\end{array} \tag{3.7}$$

where the pullback species $\mathbb{R}$ has colors defined as tuples $(R, (R_1^u, R_2^d))$ of a color $R$ of $\mathbb{S}'$ and colors $R_1$, $R_2$ of $\mathbb{S}$ such that $\psi(R_1) = \psi(R_2) = R$, and where the $n$-ary nodes of $\mathbb{R}$ are defined as pairs $(x, f)$ of a $n$-ary node $x$ of $\mathbb{S}'$ and an $n$-ary operation $f$ of $\mathcal{WCF}\,\mathbb{S}$, necessarily of the form $f = (y, 0) - \ldots - (y, n)$ for $y$ the unique $n$-ary node of $\mathbb{S}$ such that $\psi(y) = x$, by assumption that $\psi : \mathbb{S} \to \mathbb{S}'$ is injective of nodes. The canonical map of species $\mathbb{S} \to \mathbb{R}$ induced by the left-hand sides of (3.6) and (3.7) transports every color $R$ of $\mathbb{S}$ to the color $((\psi(R)^u, \psi(R)^d), R)$ and every $n$-ary node $y$ of $\mathbb{S}$ to the $n$-ary node $(\psi(y), (y, 0) - \ldots - (y, n))$ of $\mathbb{R}$. From this follows that the canonical map of species $\mathbb{S} \to \mathbb{R}$ is

injective on colors and bijective on nodes. Moreover, there are no nodes in $\mathbb{R}$ whose colors are outside of the image of $\mathbb{S}$. We conclude that the canonical functor of operads $\mathcal{F}\,\mathbb{S} \to \mathcal{F}\,\mathbb{R}$ induced by the right-hand sides of (3.6) and (3.7) is fully faithful. □

Now, let $G = (\mathcal{C}, \mathbb{S}, S, p)$ be any categorical context-free grammar, and by Prop. 3.16 consider the corresponding $\mathcal{C}$-chromatic grammar $G_{\mathsf{nodes}} = (\mathcal{C}, \phi_C\,\mathbb{S}, (A,B), p_{\mathsf{nodes}})$, where $p(S) = (A,B)$. We have a commutative diagram

$$\begin{array}{ccc} \mathcal{F}\,\mathbb{S} & \xrightarrow{\quad\mathcal{F}\,\phi_{\mathsf{colors}}\quad} & \mathcal{F}\,\phi_C\,\mathbb{S} \\ {\scriptstyle p_{\mathbb{S}}}\downarrow & & \downarrow{\scriptstyle p_{\phi_C\,\mathbb{S}}} \\ \mathcal{W}\,\mathcal{C}\,\mathcal{F}\,\mathbb{S} & \xrightarrow[\mathcal{W}\,\mathcal{C}\,\mathcal{F}\,\phi_{\mathsf{colors}}]{} & \mathcal{W}\,\mathcal{C}\,\mathcal{F}\,\phi_C\,\mathbb{S} \\ & {\scriptstyle \mathcal{W}\,q_G}\searrow \quad \mathcal{W}\,\mathcal{C} \quad \swarrow{\scriptstyle \mathcal{W}\,q_{G_{\mathsf{nodes}}}} & \end{array} \tag{3.8}$$

where the commutativity of the lower triangle follows from the equation $\phi = \phi_{\mathsf{colors}}\phi_{\mathsf{nodes}}$ and the contour / splicing adjunction. Note also that $p_{\mathsf{colors}} := \mathcal{C}\,\mathcal{F}\,\phi_{\mathsf{colors}}$ is a ULF functor of categories by Prop. 3.11 and also finitary because $\phi_{\mathsf{colors}}$ is finitary (and even finite). From this follows that $M_{\mathsf{colors}} = (\mathcal{C}\,\mathcal{F}\,\phi_C\,\mathbb{S}, \mathcal{C}\,\mathcal{F}\,\mathbb{S}, p_{\mathsf{colors}}, S^u, S^d)$ defines a finite-state automaton. By the intersection construction (Cor. 3.5) together with Lemma 3.17 and the strong translation principle (Prop. 1.11), we deduce that

$$p_{\mathsf{colors}}\,TC_{\mathbb{S},S} \;=\; TC_{\phi_C\,\mathbb{S},(A,B)} \;\cap\; \mathcal{L}_{M_{\mathsf{colors}}}.$$

Finally, using that $G$ is the image of the universal grammar $\mathsf{Univ}_{\mathbb{S},S}$ and considering the commutative diagram (3.8), we conclude:

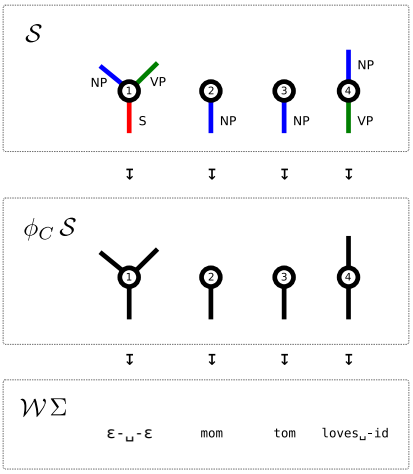
$$\mathcal{L}_G = q_G\,TC_{\mathbb{S},S} = q_{G_{\mathsf{nodes}}}\,p_{\mathsf{colors}}\,TC_{\mathbb{S},S} = q_{G_{\mathsf{nodes}}}\,(TC_{\phi_C\,\mathbb{S},(A,B)} \cap \mathcal{L}_{M_{\mathsf{colors}}}).$$

**Theorem 3.18.** *$L \subseteq \mathcal{C}(A,B)$ is a CFL of arrows iff it is a functorial image of the intersection of a $\mathcal{C}$-chromatic tree contour language with a regular language.*

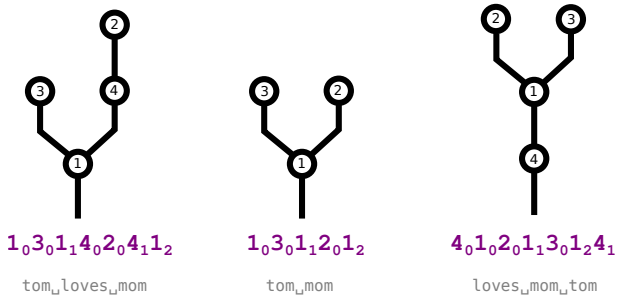
**Example 3.19.** For the original example of Figure 2, the colors-nodes factorization

$$\mathbb{S} \xrightarrow{\;\phi\;} \mathcal{W}\,\Sigma \quad = \quad \mathbb{S} \xrightarrow{\;\phi_{\mathsf{colors}}\;} \phi_C\,\mathbb{S} \xrightarrow{\;\phi_{\mathsf{nodes}}\;} \mathcal{W}\,\Sigma$$

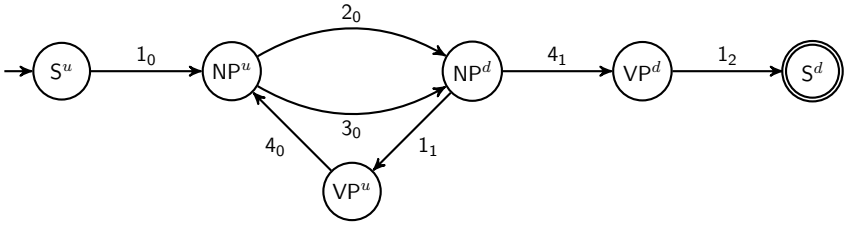
of Prop. 3.16 may be visualized as below:



Some examples of contour words in the tree contour language $TC_{\phi_C \mathbb{S},(*,*)}$ and their corresponding images under the functor $q_{G_{\mathsf{nodes}}}$ include:




Observe that only the first contour word encodes a valid parse tree mapping to a word of the language. Finally, the "coloring automaton" $M_{\mathsf{colors}}$ derived by the contour construction may be depicted by the following transition diagram:




The reader can verify that the automaton only accepts the first contour word above.

**Remark 3.20.** We view the use of contour words rather than Dyck words as a mild improvement of the original theorem by Chomsky and Schützenberger, since it removes the need for choosing arbitrarily between left brackets and right brackets to map to empty words. But there is a close relationship between tree contour words and Dyck words, and the original representation can be recovered if desired. As suggested by the right-hand side of Figure 6, there is an easy mapping from contour words to Dyck words defined by translating each corner $(x, i)$ of an $n$-ary node $x$ as follows:

$$(x,i) \mapsto \begin{cases} [x_0\ ]x_0 & i = n = 0 \\ [x_0\ [x_1 & i = 0 < n \\ ]x_i\ [x_{i+1} & 0 > i < n \\ ]x_n\ ]x_0 & 0 > i = n \end{cases}$$

This translation doubles the number of letters, and has a visual interpretation as decomposing each corner of the contour into alternating actions of walking along an edge and turning around a node. Indeed, the geometric content is so clear that we are optimistic there should be a good categorical explanation, as a way of faithfully embedding any contour category into a larger "bipartite" contour category where every arrow has been factored in two. We leave this to future work, adding that the situation appears intriguingly analogous to the embedding of the *oriented cartographic group* used to represent maps on oriented surfaces into the *unoriented cartographic group* used to represent maps on not necessarily orientable surfaces (cf. [SV90, JS94]), a link that we think is worth developing further.

3.5. **Related work.** As already mentioned, our proof that (generalized) context-free languages are closed under intersection with regular languages may be seen as a decomposition of the Bar-Hillel construction [BHPS61]. In a similar spirit, Ludmann, Pogodalla, and De Groote have recently analyzed Kanazawa's proof that languages generated by abstract categorial grammars are closed under intersection with regular languages, and shown that it defines a pullback in an appropriate category [LPdG22].

The representation theorem is one of many results appearing in Chomsky and Schützenberger's joint article on "The algebraic theory of context-free languges" [CS63]. The theorem was sketched in a technical report by Chomsky [Cho62] with attribution to their collaboration, while the first detailed proof seems to have been by Schützenberger [Sch63]. The original formulation used *two-sided*[7] Dyck languages, but would have worked just as well with ordinary (one-sided) Dyck languages and most standard treatments of the theorem now adopt that formulation [Koz97, §G]. Numerous variations of the representation theorem and extensions to other classes of languages have since been established, see [CS16] for an overview and bibliography.

The contour / splicing adjunction is fundamental to our analysis of the representation theorem, providing an unexpected geometric lens on context-free languages that is evocative of Girard's geometry of interaction. Although the adjunction is not identified, this geometric perspective is also apparent in Slavnov's recent work [Sla20] (inspired both by abstract categorial grammars and by linear logic proof-nets), wherein he constructs a compact closed monoidal category of *word cobordisms* reminiscent of the operad of spliced words.

After the original version of this article was published [MZ22], the contour / splicing adjunction was subsequently analyzed and extended by Earnshaw, Hefford, and Román, who described applications to process algebra [EHR23, Rom23]. They noticed that the splicing functor $\mathcal{W} : \mathrm{Cat} \to \mathrm{Oper}$ is closely related to a construction by Day and Street [DS04, Example 7.3], of a canonical promonoidal structure on $\mathcal{C}^{\mathrm{op}} \otimes \mathcal{C}$ for any $\mathcal{V}$-enriched category $\mathcal{C}$. We have also been further exploring the adjunction from a mathematical perspective, in ongoing work with Peter Faul.

## 4. Conclusion

We have seen that the classical notions of context-free grammar and of non-deterministic finite state automaton can be reformulated and generalized in a categorical setting, in a way that seems to have significant explanatory power. Clearly many natural questions arise in regards to integrating the deep body of work on automata theory and formal language theory within this setting, some of which we have already signalled. Another obvious question is how to define pushdown automata over categories. We find it remarkable that when one considers the nodes of a species as the production rules of a context-free grammar, the corners of its contour category correspond exactly to the *items* used in LR parsing and Earley parsing [Knu65, Ear70], which we think suggests contour categories have a role to play in analyzing left-to-right parsing algorithms from a categorical perspective. One of our original motivations for studying context-free grammars as functors and looking at parsing as a lifting problem was to develop a better understanding of the universal properties of algorithms like these, having in mind that such an understanding could then be transferred to other

[7]The two-sided Dyck language on an alphabet of $2n$ letters $x_{\pm i}$ $(1 \leq i \leq n)$ may be defined as the kernel of the canonical homomorphism $\phi : \mathrm{Mon}\langle x_{-n}, \ldots, x_{-1}, x_{+1}, \ldots, x_{+n}\rangle \to \mathrm{Grp}\langle x_1, \ldots, x_n\rangle$ from the free monoid on $2n$ generators to the free group on $n$ generators defined by $\phi(x_{+i}) = x_i$ and $\phi(x_{-i}) = x_i^{-1}$.

settings including type checking and type inference algorithms. This attempt at unification is supported by the emergence of a number of categorical and fibrational structures which seem to connect these different kinds of symbolic systems, revealing that they are founded on many of the same basic patterns.

## Acknowledgment

We thank Peter Faul for many exciting discussions of the contour / splicing adjunction, Thea Li for stimulating conversations about determinization and $\epsilon$-removal for categorical automata, and Farzad Jafarrahmani for fruitful interactions related to the material in the Addendum. We also gratefully acknowledge Bryce Clarke for many helpful insights into the contents of the paper, which improved some of the proofs, and for telling us about [Par12]. We thank Daniela Petrişan, Thomas Colcombet, and the other members of the working group on "Categories for Automata and Language Theory" at IRIF for helping us to better understand the relationship with [CP20] and with other works. We thank Benjamin Steinberg for telling us about [Jon96], which led us to [TSG88, WT86]. Finally, special thanks to Hayo Thielecke for giving a very lucid presentation of LL and LR parsing using abstract machines in 2017, which was one of our original sources of inspiration.

## References

[AL19] Benedikt Ahrens and Peter LeFanu Lumsdaine. Displayed categories. *Logical Methods in Computer Science*, 15(1), 2019. doi:10.23638/LMCS-15(1:20)2019.

[BÓ0] Jean Bénabou. Distributors at work. Notes from a course at TU Darmstadt in June 2000, taken by Thomas Streicher, 2000. URL: https://www2.mathematik.tu-darmstadt.de/~streicher/FIBR/DiWo.pdf.

[Bed88] M.A. Bednarczyk. *Categories of asynchronous systems*. PhD thesis, University of Sussex, 1988.

[BF00] Marta Bunge and Marcelo P. Fiore. Unique factorisation lifting functors and categories of linearly-controlled processes. *Mathematical Structures in Computer Science*, 10:137–163, 2000. doi:10.1017/S0960129599003023.

[BHPS61] Y. Bar-Hillel, M. Perles, and E. Shamir. On formal properties of simple phrase structure grammars. *Z. Phonetik, Sprachwissen. Kommun.*, 14(2):143–172, 1961. doi:doi:10.1524/stuf.1961.14.14.143.

[BJT97] Hans-Joachim Baues, Mamuka Jibladze, and Andy Tonks. Cohomology of monoids in monoidal categories. *Contemporary Mathematics*, 202, 1997. doi:10.1090/conm/202/02597.

[BLL98] F. Bergeron, G. Labelle, and P. Leroux. *Combinatorial Species and Tree-Like Structures*. Cambridge University Press, 1998. Translated by Margaret Readdy. doi:10.1017/CBO9781107325913.

[BM06] Nicolas Baudru and Rémi Morin. Unfolding synthesis of asynchronous automata. In Dima Grigoriev, John Harrison, and Edward A. Hirsch, editors, *Computer Science – Theory and Applications*, pages 46–57, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[BN00] Marta Bunge and Susan Niefeld. Exponentiability and single universes. *Journal of Pure and Applied Algebra*, 148:217–250, 2000. doi:10.1016/S0022-4049(98)00172-8.

[CDG+08] Hubert Comon, Max Dauchet, Rémi Gilleron, Florent Jacquemard, Denis Lugiez, Christof Löding, Sophie Tison, and Marc Tommasi. Tree Automata Techniques and Applications, 2008. URL: https://hal.inria.fr/hal-03367725.

[CE12] Bruno Courcelle and Joost Engelfriet. *Graph Structure and Monadic Second-Order Logic - A Language-Theoretic Approach*, volume 138 of *Encyclopedia of mathematics and its applications*. Cambridge University Press, 2012. doi:10.1017/CBO9780511977619.

[Cho62] Noam Chomsky. Context-free grammars and push-down storage. Quarterly Progress Report 65, Research Laboratory of Electronics, M.I.T., 1962. URL: http://hdl.handle.net/1721.1/53697.

[Con71] John Horton Conway. *Regular Algebra and Finite Machines*. Chapman and Hall, 1971. Reprinted in Dover Books on Mathematics, 2012.

[CP20] Thomas Colcombet and Daniela Petrişan. Automata minimization: a functorial approach. *Logical Methods in Computer Science*, 16(1):32:1–32:28, 2020. doi:10.23638/LMCS-16(1:32)2020.

[CS63] N. Chomsky and M. P. Schützenberger. The algebraic theory of context-free languages. In P. Braffort and D. Hirschberg, editors, *Computer Programming and Formal Systems*, volume 35 of *Studies in Logic and the Foundations of Mathematics*, pages 118–161. North-Holland, 1963. doi:10.1016/S0049-237X(08)72023-8.

[CS16] Stefano Crespi-Reghizzi and Pierluigi San Pietro. An enduring trail of language characterizations via homomorphism (talk), March 21–25 2016. Conference dedicated to the scientific legacy of M. P. Schützenberger (Bordeaux). URL: https://mps2016.labri.fr/archives/crespi.pdf.

[dG01] Philippe de Groote. Towards abstract categorial grammars. In *Association for Computational Linguistic, 39th Annual Meeting and 10th Conference of the European Chapter, Proceedings of the Conference, July 9-11, 2001, Toulouse, France*, pages 148–155. Morgan Kaufmann Publishers, 2001. doi:10.3115/1073012.1073045.

[dGP04] Philippe de Groote and Sylvain Pogodalla. On the expressive power of abstract categorial grammars: Representing context-free formalisms. *J. Log. Lang. Inf.*, 13(4):421–438, 2004. doi:10.1007/s10849-004-2114-x.

[DS02] Manfred Droste and R. M. Shortt. From petri nets to automata with concurrency. *Appl. Categorical Struct.*, 10(2):173–191, 2002. doi:10.1023/A:1014305610452.

[DS04] Brian Day and Ross Street. Quantum categories, star autonomy, and quantum groupoids. In *Galois Theory, Hopf Algebras, and Semiabelian Categories*, pages 187–225. American Mathematical Society, 2004.

[Ear70] Jay Earley. An efficient context-free parsing algorithm. *Commun. ACM*, 13(2):94–102, 1970. doi:10.1145/362007.362035.

[EHR23] Matt Earnshaw, James Hefford, and Mario Román. The produoidal algebra of process decomposition. *CoRR*, abs/2301.11867, 2023. arXiv:2301.11867, doi:10.48550/ARXIV.2301.11867.

[Eil74] Samuel Eilenberg. *Automata, Languages, and Machines: volume A*. Pure and applied mathematics. Academic Press, 1974.

[ES22] Matthew Earnshaw and Pawel Sobociński. Regular monoidal languages. In Stefan Szeider, Robert Ganian, and Alexandra Silva, editors, *47th International Symposium on Mathematical Foundations of Computer Science, MFCS 2022, August 22-26, 2022, Vienna, Austria*, volume 241 of *LIPIcs*, pages 44:1–44:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPICS.MFCS.2022.44.

[FGHW08] M. Fiore, N. Gambino, M. Hyland, and G. Winskel. The cartesian closed bicategory of generalised species of structures. *Journal of the London Mathematical Society*, 77(1):203–220, 2008. doi:10.1112/jlms/jdm096.

[FJSZ22] Uli Fahrenberg, Christian Johansen, Georg Struth, and Krzysztof Ziemianski. A Kleene theorem for higher-dimensional automata. In Bartek Klin, Slawomir Lasota, and Anca Muscholl, editors, *33rd International Conference on Concurrency Theory, CONCUR 2022, September 12-16, 2022, Warsaw, Poland*, volume 243 of *LIPIcs*, pages 29:1–29:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. URL: https://doi.org/10.4230/LIPIcs.CONCUR.2022.29, doi:10.4230/LIPICS.CONCUR.2022.29.

[GHK17] Niels Bjørn Bugge Grathwohl, Fritz Henglein, and Dexter Kozen. Infinitary axiomatization of the equational theory of context-free languages. *Fundam. Informaticae*, 150(3-4):241–257, 2017. doi:10.3233/FI-2017-1469.

[GM10] Eric Goubault and Samuel Mimram. Formal relationships between geometrical and classical models for concurrency. In Lisbeth Fajstrup, Eric Goubault, and Martin Raussen, editors, *Proceedings of the workshop on Geometric and Topological Methods in Computer Science, GETCO 2010, Aalborg, Denmark, January 11-15, 2010*, volume 283 of *Electronic Notes in Theoretical Computer Science*, pages 77–109. Elsevier, 2010. URL: https://doi.org/10.1016/j.entcs.2012.05.007, doi:10.1016/J.ENTCS.2012.05.007.

[GR62] Seymour Ginsburg and H. Gordon Rice. Two families of languages related to ALGOL. *Journal of the ACM*, 9(3):350–371, 1962. doi:10.1145/321127.321132.

[Gue20] Léonard Guetta. Polygraphs and discrete Conduché $\omega$-functors. *Higher Structures*, 4(2):134–166, 2020. doi:10.21136/HS.2020.11.

[Her00] Claudio Hermida. Representable multicategories. *Advances in Mathematics*, 151(2):164–225, 2000. doi:10.1006/aima.1999.1877.

[Her04] Claudio Hermida. Fibrations for abstract multicategories. In George Janelidze, Bodo Pareigis, and Walter Tholen, editors, *Galois Theory, Hopf Algebras, and Semiabelian Categories*, volume 43 of *Fields Institute Communications*. American Mathematical Society, 2004. doi:10.1090/fic/043/11.

[HM00] Jean-François Husson and Rémi Morin. On recognizable stable trace languages. In Jerzy Tiuryn, editor, *Foundations of Software Science and Computation Structures, Third International Conference, FOSSACS 2000, Held as Part of the Joint European Conferences on Theory and Practice of Software,ETAPS 2000, Berlin, Germany, March 25 - April 2, 2000, Proceedings*, volume 1784 of *Lecture Notes in Computer Science*, pages 177–191. Springer, 2000. doi:10.1007/3-540-46432-8_12.

[HMU07] John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. *Introduction to automata theory, languages, and computation, 3rd Edition*. Pearson international edition. Addison-Wesley, 2007.

[Hyl08] Martin Hyland. Abstract and concrete models for recursion. In O. Grumberg, T. Nipkow, and C. Pfaller, editors, *Proceedings of the NATO Advanced Study Institute on Formal Logical Methods for System Security and Correctness*, pages 175–198. IOS Press, 2008.

[Joh99] Peter Johnstone. A note on discrete Conduché fibrations. *Theory and Applications of Categories*, 5(1):1–11, 1999.

[Jon96] Peter R Jones. Profinite categories, implicit operations and pseudovarieties of categories. *Journal of Pure and Applied Algebra*, 109(1):61–95, 1996. doi:10.1016/0022-4049(95)00074-7.

[Joy81] André Joyal. Une théorie combinatoire des séries formelles. *Advances in Mathematics*, 42(1):1–82, 1981. doi:10.1016/0001-8708(81)90052-9.

[Joy86] André Joyal. Foncteurs analytiques et espèces de structures. In Gilbert Labelle and Pierre Leroux, editors, *Combinatoire énumérative*, Lecture Notes in Mathematics, pages 126–159, Berlin, Heidelberg, 1986. Springer Berlin Heidelberg. doi:10.1007/BFb0072514.

[JS94] Gareth A. Jones and David Singerman. Maps, hypermaps, and triangle groups. In L. Schneps, editor, *The Grothendieck Theory of Dessins d'Enfants*, number 200 in London Mathematical Society Lecture Note Series. Cambridge University Press, 1994. doi:10.1017/CBO9780511569302.006.

[Knu65] Donald E. Knuth. On the translation of languages from left to right. *Information and Control*, 8(6):607–639, 1965. doi:10.1016/S0019-9958(65)90426-2.

[Koz97] Dexter Kozen. *Automata and Computability*. Undergrad. texts in comp. science. Springer, 1997.

[Lam89] Joachim Lambek. Multicategories revisited. *Contemporary Mathematics*, 92:217–239, 1989. doi:10.1090/conm/092.

[Law86] F. W. Lawvere. State categories and response functors. Unpublished, May 1986.

[Lee89] René Leermakers. How to cover a grammar. In Julia Hirschberg, editor, *27th Annual Meeting of the Association for Computational Linguistics, 26-29 June 1989, University of British Columbia, Vancouver, BC, Canada, Proceedings*, pages 135–142. ACL, 1989. doi:10.3115/981623.981640.

[Lei04] Tom Leinster. *Higher Operads, Higher Categories*, volume 298 of *London Mathematical Society Lecture Note Series*. Cambridge University Press, 2004. doi:10.1017/CBO9780511525896.

[LL09] Martin Lange and Hans Leiß. To CNF or not to CNF? An efficient yet presentable version of the CYK algorithm. *Informatica Didact.*, 8, 2009. URL: https://www.informaticadidactica.de/index.php?page=LangeLeiss2009_en.

[LM10] F. W. Lawvere and M. Menni. The Hopf algebra of Möbius intervals. *Theory and Applications of Categories*, 24(10):221–265, 2010. URL: http://www.tac.mta.ca/tac/volumes/24/10/24-10abs.html.

[LPdG22] Pierre Ludmann, Sylvain Pogodalla, and Philippe de Groote. Multityped abstract categorial grammars and their composition. In Agata Ciabattoni, Elaine Pimentel, and Ruy J. G. B. de Queiroz, editors, *Logic, Language, Information, and Computation*, pages 105–122, 2022. doi:10.1007/978-3-031-15298-6_7.

[Mac98] Saunders Mac Lane. *Categories for the Working Mathematician*. Springer, 1998.

[Maz89] Antoni Mazurkiewicz. Basic notions of trace theory. In J. W. de Bakker, W. P. de Roever, and G. Rozenberg, editors, *Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency*, pages 285–363, Berlin, Heidelberg, 1989. Springer Berlin Heidelberg.

[Mel21] Paul-André Melliès. Asynchronous template games and the Gray tensor product of 2-categories. In *Proceedings of the 36th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2021, Roma, Italy, 2021*, 2021.

[MM07] Paul-André Mellies and Samuel Mimram. Asynchronous games: innocence without alternation. In *Proceedings of the 18th International Conference on Concurrency Theory, CONCUR 2007*, volume 4703 of *LNCS*, pages 395–411. Springer Verlag, 2007.

[MN24] Vincent Moreau and Lê Thành Dung Nguyên. Syntactically and semantically regular languages of lambda-terms coincide through logical relations. In *32nd EACSL Annual Conference on Computer Science Logic, CSL 2024, February 19-23, 2024, Naples, Italy*, LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024.

[Mor05] Rémi Morin. Concurrent automata vs. asynchronous systems. In Joanna Jedrzejowicz and Andrzej Szepietowski, editors, *Mathematical Foundations of Computer Science 2005, 30th International Symposium, MFCS 2005, Gdansk, Poland, August 29 - September 2, 2005, Proceedings*, volume 3618 of *Lecture Notes in Computer Science*, pages 686–698. Springer, 2005. doi:10.1007/11549345_59.

[MSS02] Martin Markl, Steve Schnider, and Jim Stasheff. *Operads in Algebra, Topology and Physics*, volume 96 of *Mathematical Surveys and Monographs*. American Mathematical Society, 2002. doi:10.1090/surv/096.

[MW67] Jorge E. Mezei and Jesse B. Wright. Algebraic automata and context-free sets. *Information and Control*, 11(1/2):3–29, 1967. doi:10.1016/S0019-9958(67)90353-1.

[MZ13] Paul-André Melliès and Noam Zeilberger. Type refinement and monoidal closed bifibrations. Unpublished, arXiv:1310.0263, October 2013. URL: https://arxiv.org/abs/1310.0263.

[MZ15] Paul-André Melliès and Noam Zeilberger. Functors are type refinement systems. In *Proceedings of the 42nd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 3–16. ACM, 2015. doi:10.1145/2676726.2676970.

[MZ16] Paul-André Melliès and Noam Zeilberger. A bifibrational reconstruction of Lawvere's presheaf hyperdoctrine. In *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science*, pages 555–564. ACM, 2016. doi:10.1145/2933575.2934525.

[MZ18] Paul-André Melliès and Noam Zeilberger. An Isbell duality theorem for type refinement systems. *Mathematical Structures in Computer Science*, 28(6):736–774, 2018. doi:10.1017/S0960129517000068.

[MZ22] Paul-André Melliès and Noam Zeilberger. Parsing as a lifting problem and the Chomsky-Schützenberger representation theorem. In *MFPS 2022 - 38th conference on Mathematical Foundations for Programming Semantics*, July 2022. doi:10.46298/entics.10508.

[Par12] Robert Paré. Mealy morphisms of enriched categories. *Applied Categorical Structures*, 20(3):251–273, 2012. doi:10.1007/S10485-010-9238-8.

[Rom23] Mario Román. *Monoidal Context Theory*. PhD thesis, Tallinn University of Technology, 2023.

[Ros95] Kimmo I. Rosenthal. Quantaloids, enriched categories and automata theory. *Applied Categorical Structures*, 3(3):279–301, 1995. doi:10.1007/BF00878445.

[RS59] Michael O. Rabin and Dana S. Scott. Finite automata and their decision problems. *IBM Journal of Research and Development*, 3(2):114–125, 1959.

[Sal09] Sylvain Salvati. Recognizability in the simply typed lambda-calculus. In *WoLLIC*, volume 5514 of *LNCS*, pages 48–60, 2009. doi:1007/978-3-642-02261-6_5.

[Sch63] M. P. Schützenberger. On context-free languages and push-down automata. *Information and control*, 6(3):246–264, 1963.

[Sch15] Gilles Schaeffer. Planar maps. In Miklós Bóna, editor, *Handbook of Enumerative Combinatorics*, pages 335–396. CRC, 2015. URL: http://www.lix.polytechnique.fr/~schaeffe/Master/HB.pdf.

[Shu] Michael Shulman. LNL polycategories and doctrines of linear logic. *Logical Methods in Computer Science*, 19(2):1:1–1:54. doi:10.46298/lmcs-19(2:1)2023.

[Sip13] Michael Sipser. *Introduction to the theory of computation, 3rd edition*. Cengage Learning, 2013.

[Sla20] Sergey Slavnov. Classical linear logic, cobordisms and categorial grammars, 2020. arXiv:1911.03962. URL: https://arxiv.org/abs/1911.03962.

[SMFK91] Hiroyuki Seki, Takashi Matsumura, Mamoru Fujii, and Tadao Kasami. On multiple context-free grammars. *Theoretical Computer Science*, 88(2):191–229, 1991. doi:https://doi.org/10.1016/0304-3975(91)90374-B.

[SS88] Seppo Sippu and Eljas Soisalon-Soininen. *Parsing Theory - Volume I: Languages and Parsing*, volume 15 of *EATCS Monographs on Theoretical Computer Science*. Springer, 1988. doi:10.1007/978-3-642-61345-6.

[Ste01] Benjamin Steinberg. Finite state automata: a geometric approach. *Transactions of the American Mathematical Society*, 353(9):3409–3464, 2001.

[Str96] Ross Street. Categorical structures. In M. Hazewinkel, editor, *Handbook of Algebra*, volume 1, pages 529–577. North-Holland, 1996.

[SV90] G. Shabat and V. Voevodsky. Drawing curves over number fields. In P. Cartier, Luc Illusie, Nicholas M. Katz, Gérard Laumon, Yuri I. Manin, and Kenneth A. Ribet, editors, *The Grothendieck festschrift III*, number 88 in Progress in Mathematics, pages 199–227. Birkhäuser, 1990. URL: https://www.math.ias.edu/vladimir/sites/math.ias.edu.vladimir/files/drawing_curves_published.pdf.

[TSG88] Denis Thérien and Małgorzata Sznajder-Glodowski. Finite categories and regular languages. In *Mathematical Problems in Computation Theory*, volume 21 of *Banach Center Publications*, pages 395–402. 1988.

[Wal89] R. F. C. Walters. A note on context-free languages. *Journal of Pure and Applied Algebra*, 62(2):199–203, 1989. doi:10.1016/0022-4049(89)90151-5.

[WT86] Alex Weiss and Denis Thérien. Varieties of finite categories. *Informatique théorique et applications*, 20(3):357–366, 1986.

[Zie87] Wieslaw Zielonka. Notes on finite asynchronous automata. *RAIRO – Theoretical Informatics and Applications*, 21:99–135, 1987.

## Addendum A. gCFLs as initial models of gCFGs

We begin to explore here another more abstract perspective on (generalized) context-free grammars and context-free languages, under which a CFG defines a kind of theory that generates a CFL as its *initial model*. On the one hand, this viewpoint encompasses the old idea of treating CFGs as defining certain systems of polynomial equations and CFLs as minimal solutions to those systems of equations (cf. [GR62, MW67], [Con71, Ch. 10], [GHK17]). At the same time, our categorical formulation is in the broad spirit of functorial semantics of algebraic theories à la Lawvere, adapted to an operadic and fibrational setting. We will see that one benefit of this perspective is that it enables us to consider a more expansive notion of "language" generated by a grammar, recovering the traditional notion of formal language as subset of words (or subset of arrows/constants) as a special case.

Before stating the general definitions, let us give an example. The Dyck language of well-bracketed words over an alphabet $\Sigma$ containing a pair of brackets '[' and ']' may be generated by either of the following context-free grammars with a single non-terminal:

$$G_1 \;=\; \begin{array}{lcl} S & \to & \epsilon \\ S & \to & [S] \\ S & \to & SS \end{array} \qquad G_2 \;=\; \begin{array}{lcl} S & \to & \epsilon \\ S & \to & [S]S \end{array} \tag{A.1}$$

Although the language $L = \mathcal{L}_{G_1} = \mathcal{L}_{G_2}$ generated by both grammars is the same, $G_1$ and $G_2$ may be seen as implicitly stating two different equations *satisfied* by this language,

$$L = \epsilon + [L] + LL \tag{A.2}$$

$$L = \epsilon + [L]L \tag{A.3}$$

where we have written $+$ for union (and implicitly make use of the spliced concatenation of languages as defined in Prop. 1.9(2)). Equation (A.2) has other solutions besides the

language of Dyck words, such as the total language $L = \Sigma^*$, while interestingly (A.3) only admits a unique solution. However, in both cases one can easily establish that the context-free language generated by the grammar provides the unique *minimal* solution to the corresponding equation, in the sense that it is included in any other solution.

To describe the proper abstract setting for such systems of equations, we need to introduce here some fibrational concepts for functors of operads, for which we again find it useful to adapt some conventions from our type refinement systems work. Extending the notation from §1.6, we write $\alpha : R_1, \ldots, R_n \Rightarrow^q_f R$ to indicate that $\alpha : R_1, \ldots, R_n \to R$ is an operation in $\mathcal{E}$ with image $q(\alpha) = f$ in $\mathcal{B}$, relative to a functor of operads $q : \mathcal{E} \to \mathcal{B}$. We sometimes omit the superscript and write $\alpha : R_1, \ldots, R_n \Rightarrow_f R$ if $q$ is clear from context. We write $\mathcal{E}_f(R_1, \ldots, R_n; R)$ for the set of operations $\{\, \alpha \mid \alpha : R_1, \ldots, R_n \Rightarrow^q_f R \,\}$.

**Definition A.1.** A **cone** in an operad $\mathcal{O}$ is a family of operations $(g_i : \Delta_i \to A)_{i \in I}$ in $\mathcal{O}$ with the same output color $A$.

**Definition A.2.** Let $q : \mathcal{E} \to \mathcal{B}$ be a functor of operads. A cone $(\alpha_i : \Omega_i \Rightarrow^q_{g_i} R)_{i \in I}$ in $\mathcal{E}$ is said to be **minimal** over a cone $(g_i : \Delta_i \to A)_{i \in I}$ in $\mathcal{B}$ (relative to $q$) if for every operation $f : \Gamma, A, \Gamma' \to B$ in $\mathcal{B}$ with $|\Gamma| = k$, the function

$$(- \circ_k \alpha_i)_{i \in I} \quad : \quad \mathcal{E}_f(\Theta, R, \Theta'; S) \longrightarrow \prod_{i \in I} \mathcal{E}_{f \circ_k g_i}(\Theta, \Omega_i, \Theta'; S)$$

induced by precomposition with the elements of the cone is invertible. In this case, we say that $R$ equipped with the cone $(\alpha_i : \Omega_i \Rightarrow^q_{g_i} R)_{i \in I}$ is a **minimal lift** of the cone $(g_i)_{i \in I}$ to the family of lists of colors $(\Omega_i \sqsubset^q \Delta_i)_{i \in I}$ refining the inputs of the $g_i$'s.

Equivalently, $(\alpha_i : \Omega_i \Rightarrow^q_{g_i} R)_{i \in I}$ is a minimal cone when for every operation $f$ as above and any cone $(\beta_i : \Theta, \Omega_i, \Theta' \Rightarrow^q_{f \circ_k g_i} S)_{i \in I}$ over the cone $(f \circ_k g_i : \Gamma, \Delta_i, \Gamma' \to B)_{i \in I}$, there exists a unique operation $\bar{\beta} : \Theta, R, \Theta' \Rightarrow^q_f S$ such that $\beta_i = \bar{\beta} \circ_k \alpha_i$ for all $i \in I$.

In the special case of a singleton family, an operation $\alpha : \Omega \Rightarrow_g R$ of $\mathcal{E}$ is a minimal cone just in case it is (strongly) opcartesian in the sense of Hermida relative to a functor of multicategories [Her00, Her04], which extends the classical notion of opcartesian arrow relative to a functor of categories. In this case, we say that $R$ is the **pushforward** of $\Omega$ along $g$, generalizing the image of a subset along a function. On the other hand, in the special case of a minimal cone $(\alpha_i : R_i \Rightarrow_{id_B} R)_{i \in I}$ of operations in $\mathcal{E}$ all lying over the same identity operation in $\mathcal{B}$, we say that $R$ is the **fiberwise coproduct** of the $R_i$, generalizing the union of subsets of a set. This means in particular that we have an isomorphism

$$\mathcal{E}_f(\Theta, R, \Theta'; S) \cong \prod_{i \in I} \mathcal{E}_f(\Theta, R_i, \Theta'; S)$$

for every compatible operation $f$. We briefly considered minimal cones in the same sense (relative to a functor of categories) in [MZ13], there referred to as "weighted union types". Definition A.2 is also in a similar spirit to the very general notion of "extremal cone" recently introduced by Shulman in his work on LNL polycategories and doctrines of linear logic [Shu], which inspired our terminology.

Given a cone $(g_i : \Gamma_i \to A)_{i \in I}$ and a family of lists of colors $\Omega_i \sqsubset \Gamma_i$ refining the inputs of the $g_i$'s, there exists at most one minimal lift $(\alpha_i : \Omega_i \Rightarrow^q_{g_i} R)_{i \in I}$ of $(g_i)_{i \in I}$ to $(\Omega_i)_{i \in I}$, up to canonical isomorphism. Indeed, if $(\alpha_i : \Omega_i \Rightarrow_{g_i} R)_{i \in I}$ and $(\alpha'_i : \Omega_i \Rightarrow_{g_i} R')_{i \in I}$ are two minimal lifts, the universal property immediately implies there is a unique isomorphism

$\bar{\beta} : R \cong R'$ in $\mathcal{E}$ lying over the identity $id_A$ in $\mathcal{B}$ such that $\alpha'_i = \bar{\beta} \circ \alpha_i$ for all $i \in I$. We indicate the presence of such a canonical isomorphism by $R \equiv R'$, and write $\sum_{i\in I} g_i \, \Omega_i$ for some choice of object $R$ coming together with a minimal cone

$$(in_j : \Omega_j \underset{g_j}{\Longrightarrow} \sum_{i\in I} g_i \, \Omega_i)_{j\in I}$$

The notation is motivated by the fact that if the functor $q : \mathcal{E} \to \mathcal{B}$ has sufficient fibrational structure, then the minimal lift may be decomposed as a fiberwise coproduct of pushforwards (cf. [MZ13, p.13], [Shu, Thm. 4.28]).

**Proposition A.3.** *Let $q : \mathcal{E} \to \mathcal{B}$ be a functor of operads. The following are equivalent:*

(1) *There is a minimal lift $\sum_{i\in I} g_i \, \Omega_i \sqsubset A$ of every cone $(g_i : \Gamma_i \to A)_{i\in I}$ in $\mathcal{B}$ to any family of lists of colors $\Omega_i \sqsubset \Gamma_i$ in $\mathcal{E}$.*
(2) *$q$ has all pushforwards and fiberwise coproducts, i.e., for any operation $g : \Gamma \to A$ and list of colors $\Omega \sqsubset \Gamma$ there is a pushforward $g \, \Omega \sqsubset A$, and for any family of colors $(R_i \sqsubset A)_{i\in I}$, there is a fiberwise coproduct $\sum_{i\in I} R_i \sqsubset A$.*

*Moreover, the equivalence holds while maintaining any bound $|I| < \kappa$ on the cardinalities of the indexing sets of the cones $(g_i)_{i\in I}$ in (1) and the families of colors $(R_i)_{i\in I}$ in (2).*

*Proof.* The implication (1) $\Rightarrow$ (2) is immediate. For the implication (2) $\Rightarrow$ (1), we build the minimal lift of $(g_i : \Gamma_i \to A)_{i\in I}$ to $(\Omega_i \sqsubset \Gamma_i)_{i\in I}$ by constructing the family of pushforwards $(R_i = g_i \, \Omega_i \sqsubset A)_{i\in I}$ and taking their fiberwise coproduct $\sum_{i\in I} R_i \sqsubset A$. Since the indexing set $I$ is left unchanged, existence of minimal lifts of cones of cardinality $< \kappa$ is equivalent to existence of pushforwards and finite fiberwise coproducts of cardinality $< \kappa$. □

When either of the equivalent conditions of Prop. A.3 holds with $\kappa = \omega$, and moreover $q$ comes equipped with a choice of minimal lifts, then we say that $q$ **has polynomials**, or is a "polynomial refinement system". The terminology is motivated by the fact that the set of colors of $\mathcal{E}$ is closed under taking finite sums of monomials weighted by operations of $\mathcal{B}$, and moreover that such polynomial expressions satisfy expected laws up to canonical isomorphism.

**Proposition A.4.** *The following identities hold*

$$\sum_{i\in I} \sum_{j\in J} R_{ij} \equiv \sum_{(i,j)\in I\times J} R_{ij} \tag{A.4}$$

$$f(\Theta, \sum_{i\in I} R_i, \Theta') \equiv \sum_{i\in I} f(\Theta, R_i, \Theta') \tag{A.5}$$

$$f(g_1 \, \Omega_1, \ldots, g_n \, \Omega_n) \equiv (f \circ (g_1, \ldots, g_n))(\Omega_1, \ldots, \Omega_n) \tag{A.6}$$

$$id(R) \equiv R \tag{A.7}$$

*in the sense that whenever the minimal lift on one side exists then so does the other, with a canonical isomorphism between them.*

We are now ready to define models of generalized context-free grammars and explain the sense in which an initial model provides a minimal solution to a system of polynomial equations. The definition depends only on the grammar's underlying functor $p : \mathcal{F}\,\mathbb{S} \to \mathcal{O}$, which we refer to as a bare gCFG (analogous to the bare NDFA we considered in §2).

**Definition A.5.** A **model** of an arbitrary functor of operads $p : \mathcal{D} \to \mathcal{O}$ inside another functor $q : \mathcal{E} \to \mathcal{B}$ is just a commuting square of functors:

$$\begin{array}{ccc} \mathcal{D} & \xrightarrow{\tilde{M}} & \mathcal{E} \\ {\scriptstyle p}\downarrow & & \downarrow{\scriptstyle q} \\ \mathcal{O} & \xrightarrow{M} & \mathcal{B} \end{array} \tag{A.8}$$

By extension, we say that $(M, \tilde{M})$ is a model of a gCFG $G$ if it is a model of its underlying bare gCFG $p : \mathcal{F}\ \mathbb{S} \to \mathcal{O}$. Note that the commutative square (A.8) just says that the pair of functors $(M, \tilde{M})$ form a *morphism of functors* from $p$ to $q$, or a *morphism of refinement systems* in the terminology of [MZ18]. But sometimes we want more of a model of a gCFG, namely that the morphism $(M, \tilde{M}) : p \to q$ picks out a selection of minimal cones in the following sense.

**Definition A.6.** A model $(M, \tilde{M})$ of a bare gCFG $p : \mathcal{F}\ \mathbb{S} \to \mathcal{O}$ (with associated map of species $\phi : \mathbb{S} \to \mathcal{O}$) in $q : \mathcal{E} \to \mathcal{B}$ is said to be **exact** if for every color $R$ of $\mathbb{S}$, the cone of nodes in $\mathbb{S}$ with output color $R$

$$\mathbb{S}_R = (x : R_1, \ldots, R_k \xRightarrow[g]{\phi} R)_{x \in \mathbb{S}}$$

is mapped to a $q$-minimal cone in $\mathcal{E}$

$$\tilde{M}\mathbb{S}_R = (\tilde{M}x : \tilde{M}R_1, \ldots, \tilde{M}R_k \xRightarrow[Mg]{q} \tilde{M}R)_{x \in \mathbb{S}}.$$

Thus, if we express the existence of minimal cones using the sum-of-pushforward notation introduced above, an exact model $(M, \tilde{M})$ of a gCFG corresponds to a solution for the system of equations

$$\tilde{M}R \equiv \sum_{R_1, \ldots, R_k \Rightarrow_g^\phi R} Mg(\tilde{M}R_1, \ldots, \tilde{M}R_k) \tag{A.9}$$

with one such equation for each color of $\mathbb{S}$. Let us emphasize that although such a solution interprets each non-terminal as the apex of a minimal cone, the solution itself is not necessarily minimal globally. However, we will establish as Proposition A.9 below that under good conditions (namely, when $q$ has polynomials), if $(M, \tilde{M})$ is the *initial model* of $p$ in $q$, then it is also exact and therefore the initial exact model. To make this statement precise, we first have to define what are morphisms of models.

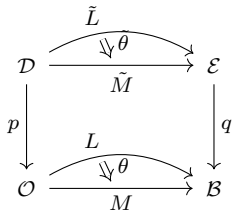
Recall that given two functors of operads $L, M : \mathcal{O} \to \mathcal{P}$ with the same source and target, a *natural transformation* $\theta : L \Rightarrow M$ is defined as a family of unary operations $\theta_R : LR \to MR$ in $\mathcal{P}$ such that

$$Mf \circ (\theta_{R_1}, \ldots, \theta_{R_n}) = \theta_R \circ Lf \quad : \quad LR_1, \ldots, LR_n \longrightarrow MR \tag{A.10}$$

for every operation $f : R_1, \ldots, R_n \to R$ in $\mathcal{O}$.

**Definition A.7.** Let $(L, \tilde{L})$ and $(M, \tilde{M})$ be models of $p$ in $q$. A **morphism of models** $(L, \tilde{L}) \Longrightarrow (M, \tilde{M})$ is given by a pair of natural transformations $\theta : L \Rightarrow M$ and $\tilde{\theta} : \tilde{L} \Rightarrow \tilde{M}$

such that the diagram below commutes




(A.11)

in the sense that the natural transformations obtained by whiskering are equal $q \circ \tilde{\theta} = \theta \circ p$.

In other words, a morphism of models is a *2-morphism* $(\theta, \tilde{\theta}) : (L, \tilde{L}) \Longrightarrow (M, \tilde{M}) : p \to q$. We define morphisms of exact models in the same way – that is, we do not impose any extra conditions. Given two functors of operads $p$ and $q$, we write $\mathrm{Mod}(p, q)$ for the category of models of $p$ in $q$ and 2-morphisms between them. Similarly, when $p$ is a bare gCFG, we write $\mathrm{ExMod}(p, q)$ for the full subcategory of $\mathrm{Mod}(p, q)$ spanned by the exact models.

Definitions A.5 and A.6 do not require the target $q : \mathcal{E} \to \mathcal{B}$ to have polynomials. However, when it does, models of a bare gCFG $p : \mathcal{F}\ \mathbb{S} \to \mathcal{O}$ admit an equivalent characterization as *algebras of an endofunctor*

$$(-)^p : \mathrm{DiscMod}(p, q) \to \mathrm{DiscMod}(p, q)$$

on the category of *discrete models* of $p$ in $q$, defined by $\mathrm{DiscMod}(p, q) := \mathrm{Mod}(p \circ \mathit{incl}, q)$, where $\mathit{incl} : \mathcal{F}\ |\mathbb{S}| \to \mathcal{F}\ \mathbb{S}$ is the inclusion from the free operad over the discrete species $|\mathbb{S}|$ with same set of colors as $\mathbb{S}$ and no nodes. Note that a discrete model $(M, \tilde{M})$ is the data of a functor $M : \mathcal{O} \to \mathcal{B}$ together with a family of colors $\tilde{M}R \sqsubset^q MA$ for every color $R \sqsubset^p A$ of $\mathbb{S}$.

For any discrete model $(M, \tilde{M})$ of a bare gCFG $p$ in $q$, we can construct a new discrete model $(M, \tilde{M}^p)$ where the family of colors $\tilde{M}^p R$ is defined by the sum of pushforwards:

$$\tilde{M}^p R = \sum_{R_1, \ldots, R_k \Rightarrow_g^\phi R} Mg(\tilde{M}R_1, \ldots, \tilde{M}R_k)$$

This defines an endofunctor on $\mathrm{DiscMod}(p, q)$ since the pushforward and fiberwise coproduct are functorial. Moreover, by precomposition with $\mathit{incl}$, every ordinary model $(M, \tilde{M})$ of $p$ in $q$ has an underlying discrete model (which we will also denote $(M, \tilde{M})$ for ease of notation), equipped with a canonical $(-)^p$-algebra structure given by the family of unary operations

$$\alpha_R^{\tilde{M}} : \tilde{M}^p R \Rightarrow \tilde{M}R$$

obtained by applying the universal property of the sum-of-pushforwards to the cone of operations $\tilde{M}\mathbb{S}_R$.

**Proposition A.8.** *The assignment of a $(-)^p$-algebra structure $\alpha^{\tilde{M}} : \tilde{M}^p \Rightarrow \tilde{M}$ to any model $(M, \tilde{M})$ of a bare gCFG $p$ in a polynomial refinement system $q$ induces an isomorphism*

$$\mathrm{Mod}(p, q) \cong (-)^p\text{-Alg}$$

*between the category of models of $p$ in $q$ and the category of algebras for the endofunctor $(-)^p : \mathrm{DiscMod}(p, q) \to \mathrm{DiscMod}(p, q)$. This restricts to an isomorphism*

$$\mathrm{ExMod}(p, q) \cong (-)^p\text{-Fix}$$

*between the subcategory of exact models of $p$ in $q$ and the subcategory of $(-)^p$-algebras with an invertible algebra structure.*

Since the mapping $(M, \tilde{M}) \mapsto (M, \tilde{M}^p)$ preserves the underlying interpretation $M : \mathcal{O} \to \mathcal{B}$ of the base operad, it in fact defines a family of endofunctors

$$(-)^p_M : \mathrm{DiscMod}_M(p, q) \to \mathrm{DiscMod}_M(p, q)$$

where we write $\mathrm{DiscMod}_M(p, q)$ for the subcategory of discrete models of $p$ in $q$ with underlying base interpretation $M$, and with morphisms between them such that the natural transformation in the bottom part of (A.11) is the identity $\theta = id_M$. We similarly write $\mathrm{Mod}_M(p, q)$ and $\mathrm{ExMod}_M(p, q)$ for the corresponding subcategories of models and exact models with base interpretation $M$. It follows from Proposition A.8 that we have isomorphisms:

$$\mathrm{Mod}_M(p, q) \cong (-)^p_M\text{-Alg} \tag{A.12}$$

$$\mathrm{ExMod}_M(p, q) \cong (-)^p_M\text{-Fix} \tag{A.13}$$

We can use these alternative characterizations of models of a gCFG to immediately derive the existence of initial exact models (globally, or with a given base interpretaion $M$) from the existence of initial models.

**Proposition A.9.** *For every bare gCFG $p$ and polynomial refinement system $q$, any initial object in* $\mathrm{Mod}(p, q)$ *(respectively, in* $\mathrm{Mod}_M(p, q)$*) is also an initial object in* $\mathrm{ExMod}(p, q)$ *(resp.* $\mathrm{ExMod}_M(p, q)$*).*

*Proof.* By the previous proposition, if $(M, \tilde{M})$ is initial in $\mathrm{Mod}(p, q)$ then $\alpha^{\tilde{M}} : \tilde{M}^p \Rightarrow \tilde{M}$ is an initial $(-)^p$-algebra. By Lambek's Lemma, $\alpha^{\tilde{M}}$ is invertible and hence $(M, \tilde{M})$ is exact, and therefore an initial object of $\mathrm{ExMod}(p, q)$ since it is a full subcategory of $\mathrm{Mod}(p, q)$. The same argument works even if $\tilde{M}$ is only initial seen as an object $(M, \tilde{M})$ of $\mathrm{Mod}_M(p, q)$. □

Proposition A.9 motivates us to introduce the following terminology.

**Definition A.10.** Let $G$ be a gCFG with underlying bare gCFG $p : \mathcal{F}\ \mathbb{S} \to \mathcal{O}$ and start symbol $S \sqsubset^p A$. Let $q : \mathcal{E} \to \mathcal{B}$ be a polynomial refinement system, and suppose that there is an initial model $(M, \tilde{M})$ of $p$ in $q$ (which is hence unique up to canonical isomorphism by universality, and an exact model by Prop. A.9). We then speak of the **$q$-language** of $G$, written $\mathcal{L}^q_G$, as the interpretation $\tilde{M}\ S \sqsubset^q M\ A$ of the start symbol in this initial model. Similarly, given a base interpretation functor $M : \mathcal{O} \to \mathcal{B}$, if $\tilde{M}$ is initial in $\mathrm{Mod}_M(p, q)$, then we speak of the **$q$-language relative to** $M$ of $G$, written $\mathcal{L}^{M,q}_G$.

We will see that the language of constants of a gCFG in the sense of Def. 1.15 may be recovered as a particular case of $q$-language, for $q = \mathsf{sub} : \mathrm{Subset} \to \mathrm{Set}$ defined below. However, every gCFG also generates another $q$-language that is in a sense more canonical, as we now explain.

For any operad $\mathcal{P}$, there is an operad $\mathcal{P}^{\to}$ whose colors are given by the unary operations $u$ of $\mathcal{P}$, and whose $n$-ary operations $u_1, \ldots, u_n \to u$ are given by pairs $(f_s, f_t)$ of $n$-ary operations of $\mathcal{P}$ such that $f_t \circ (u_1, \ldots, u_n) = u \circ f_s$. This operad comes equipped with two evident functors $\mathsf{src}, \mathsf{tgt} : \mathcal{P}^{\to} \to \mathcal{P}$ defined by first and second projection. In the case $\mathcal{P} = \mathrm{Set}$, the operad $\mathrm{Set}^{\to}$ has unary functions $f : Y \to X$ as colors.

**Proposition A.11.** $\mathsf{tgt} : \mathrm{Set}^{\to} \to \mathrm{Set}$ *has polynomials, with pushforwards and fiberwise coproducts constructed as follows:*

- *Let* $u_1 : Y_1 \to X_1, \dots, u_n : Y_n \to X_n$ *be an* $n$*-tuple of unary functions, and let* $f : X_1, \dots, X_n \to X$ *be an* $n$*-ary function. The pushforward of* $(u_1, \dots, u_n)$ *along* $f$ *is given by their parallel composition* $f \circ (u_1, \dots, u_n)$*, interpreted as a unary function from the cartesian product* $Y_1 \times \cdots \times Y_n \to X$*.*
- *Let* $(u_i : Y_i \to X)_{i \in I}$ *be a collection of unary functions into the same set. Their fiberwise coproduct is given by the cotupling* $[u_i]_{i \in I} : \coprod_{i \in I} Y_i \to X$*.*

Now, every operad $\mathcal{O}$ comes equipped with a canonical functor

$$\mathrm{con}_{\mathcal{O}} : \mathcal{O} \to \mathrm{Set}$$

(we omit the subscript when clear from context) defined on colors by taking $\mathrm{con}\, A = \mathcal{O}(\cdot; A) = \{\, c \mid c : A \,\}$ to be the set of constants of color $A$ in $\mathcal{O}$, and on operations $f : A_1, \dots, A_n \to B$ by taking $\mathrm{con}\, f$ to be the function $(c_1, \dots, c_n) \mapsto f \circ (c_1, \dots, c_n)$ that sends any $n$-tuple of constants to the constant defined by parallel composition into the inputs of $f$. For example, in the case of an operad of spliced arrows, $\mathrm{con} : \mathcal{WC} \to \mathrm{Set}$ sends a color $(A, B)$ to the homset $\mathcal{C}(A, B)$, and an operation $w_0 - \dots - w_n : (A_1, B_1), \dots, (A_n, B_n) \to (A, B)$ to the function

$$\mathrm{con}(w_0 - \dots - w_n) : \mathcal{C}(A_1, B_1) \times \cdots \times \mathcal{C}(A_n, B_n) \to \mathcal{C}(A, B)$$

that maps an $n$-tuple of arrows $(u_1, \dots, u_n)$ to their spliced composition $w_0 u_1 w_1 \dots u_n w_n$. A functor $\mathcal{O} \to \mathrm{Set}$ is also called an "operad algebra", and an important property of $\mathrm{con}_{\mathcal{O}}$ is that it is the initial $\mathcal{O}$-algebra, in the sense that it has a unique natural transformation $!_M : \mathrm{con}_{\mathcal{O}} \Rightarrow M$ to any other functor $M : \mathcal{O} \to \mathrm{Set}$. This natural transformation is defined by the family of functions $\mathrm{con}\, A \to MA$ sending a constant $c : A$ of $\mathcal{O}$ to the element $Mc$ of $MA$ determined by the algebra structure.

For any functor of operads $p : \mathcal{D} \to \mathcal{O}$, we can therefore build a triangle

$$\begin{array}{ccc} \mathcal{D} & \xrightarrow{\mathrm{con}_{\mathcal{D}}} & \mathrm{Set} \\ {\scriptstyle p}\downarrow & \swarrow_{\mathrm{con}_p} & \nearrow_{\mathrm{con}_{\mathcal{O}}} \\ \mathcal{O} & & \end{array} \tag{A.14}$$

where the natural transformation $\mathrm{con}_p = !_{\mathrm{con}_{\mathcal{O}} \circ p}$ is uniquely determined by initiality of $\mathrm{con}_{\mathcal{D}}$. Concretely, $\mathrm{con}_p$ is given by the family of functions sending any constant in $\mathcal{D}$ to the corresponding constant in $\mathcal{O}$ obtained by mapping along the functor $p$. Now, in general, giving a natural transformation $\theta : L \Rightarrow M : \mathcal{O} \to \mathcal{P}$ is equivalent to giving a functor $\tilde{\theta} : \mathcal{O} \to \mathcal{P}^{\to}$ such that $\mathsf{src} \circ \tilde{\theta} = L$ and $\mathsf{tgt} \circ \tilde{\theta} = M$. In particular, the natural transformation (A.14) induces a functor $\widetilde{\mathrm{con}}_p : \mathcal{D} \to \mathrm{Set}^{\to}$ making the below square commute:

$$\begin{array}{ccc} \mathcal{D} & \xrightarrow{\widetilde{\mathrm{con}}_p} & \mathrm{Set}^{\to} \\ {\scriptstyle p}\downarrow & & \downarrow{\scriptstyle \mathsf{tgt}} \\ \mathcal{O} & \xrightarrow{\mathrm{con}_{\mathcal{O}}} & \mathrm{Set} \end{array} \tag{A.15}$$

**Proposition A.12.** *The morphism* $(\mathrm{con}_{\mathcal{O}}, \widetilde{\mathrm{con}}_p) : p \to \mathsf{tgt}$ *is an initial object in* $\mathrm{Mod}(p, \mathsf{tgt})$*.*

*Proof.* Let $M : \mathcal{O} \to \mathrm{Set}$ and $\tilde{M} : \mathcal{D} \to \mathrm{Set}^{\to}$ be any pair of functors forming a morphism $(M, \tilde{M}) : p \to \mathsf{tgt}$. A 2-morphism $(\theta, \tilde{\theta}) : (\mathrm{con}_{\mathcal{O}}, \widetilde{\mathrm{con}}_p) \Rightarrow (M, \tilde{M}) : p \to \mathsf{tgt}$ is given by a pair of natural transformations $\theta : \mathrm{con}_{\mathcal{O}} \Rightarrow M$ and $\tilde{\theta} : \widetilde{\mathrm{con}}_p \Rightarrow \tilde{M}$ such that $\mathsf{tgt} \circ \tilde{\theta} = \theta \circ p$, or

equivalently a pair $\theta : \mathrm{con}_{\mathcal{O}} \Rightarrow M$ and $\rho : \mathrm{con}_{\mathcal{D}} \Rightarrow \mathsf{src} \circ \tilde{M}$ which combined with the natural transformations $\mathrm{con}_{\mathcal{D}} \Rightarrow p \circ \mathrm{con}_{\mathcal{O}}$ and $\mathsf{src} \circ \tilde{M} \Rightarrow p \circ M$ form a family of commutative squares

$$\begin{array}{ccc} \mathrm{con}\, R & \xrightarrow{\rho_R} & \mathsf{src}(\tilde{M} R) \\ \downarrow & & \downarrow \\ \mathrm{con}\, A & \xrightarrow{\theta_A} & MA \end{array} \tag{A.16}$$

in Set for every color $R \sqsubset A$ of $\mathcal{D}$. But such a pair of natural transformations $\theta$ and $\rho$ exist uniquely by initiality of $\mathrm{con}_{\mathcal{O}}$ and $\mathrm{con}_{\mathcal{D}}$ respectively, while the square (A.16) commutes by initiality of $\mathrm{con}_{\mathcal{D}}$ since it is a composite of natural transformations. □

**Corollary A.13.** *Every bare gCFG generates an initial model in* $\mathsf{tgt} : \mathrm{Set}^{\to} \to \mathrm{Set}$.

In the case of a categorical CFG, the initial model in $\mathsf{tgt}$ interprets every non-terminal $R \sqsubset (A, B)$ as the set $\mathrm{con}\, R$ of parse trees with root label $R$, equipped with the function

$$\mathrm{con}_p\, R \quad : \quad \mathrm{con}_{\mathcal{F}\,\mathbb{S}}\, R \longrightarrow \mathcal{C}(A, B)$$

mapping a tree $\alpha : R$ to the underlying arrow $p(\alpha) \in \mathcal{C}(A, B)$ that it derives. In particular, the language $\mathcal{L}_G^{\mathsf{tgt}} = \mathrm{con}(S) \to \Sigma^*$ generated by an ordinary context-free grammar may be considered as a kind of "proof-relevant" language, in the sense that it encodes not just a subset of words generated by $G$ but also the set of parse trees of every word in the language. Notably, a given word may have more than one derivation, and all of those distinct derivations are incorporated into its $\mathsf{tgt}$-language.

To recover the language generated by a grammar in the traditional sense, consider next the operad Subset whose colors are pairs $(X, U)$ of a set $X$ and of a subset $U \subseteq X$, and whose operations $f : (X_1, U_1), \ldots, (X_n, U_n) \to (X, U)$ are functions $f : X_1 \times \cdots \times X_n \to X$ such that

$$x_1 \in U_1, \ldots, x_n \in U_n \text{ implies } f(x_1, \ldots, x_n) \in U.$$

Subset comes equipped with an evident forgetful functor $\mathsf{sub} : \mathrm{Subset} \to \mathrm{Set}$ sending any subset $(X, U)$ to its base $X$.

**Proposition A.14.** $\mathsf{sub}$ *has pushforwards and fiberwise coproducts, given respectively by taking the image of an* $n$*-tuple of subsets* $R_1 \subseteq X_1, \ldots, R_n \subseteq X_n$ *along an* $n$*-ary function* $f : X_1, \ldots, X_n \to X$ *to define the subset* $f(R_1, \ldots, R_n) \subseteq X$*, and by taking the union* $\bigcup_{i \in I} R_i \subseteq X$ *of a collection of subsets* $(R_i \subseteq X)_{i \in I}$ *of the same set. In particular,* $\mathsf{sub}$ *has polynomials.*

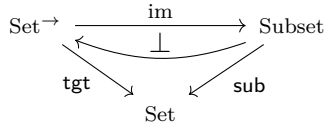
Moreover, there is a morphism of refinement systems from $\mathsf{tgt}$ to $\mathsf{sub}$,

$$\begin{array}{ccccc} \mathrm{Set}^{\to} & & \xrightarrow{\quad\mathrm{im}\quad} & & \mathrm{Subset} \\ & \searrow^{\mathsf{tgt}} & & \swarrow_{\mathsf{sub}} & \\ & & \mathrm{Set} & & \end{array}$$

given by the functor $\mathrm{im} : \mathrm{Set}^{\to} \to \mathrm{Subset}$ which sends any function $u : X \to Y$ to its image $u(X) \subseteq Y$, and every pair $(f_s, f_t)$ of functions $f_s : X_1 \times \cdots \times X_n \to X$ and $f_t : Y_1 \times \cdots \times Y_n \to Y$ defining an operation $u_1, \ldots, u_n \to u$ of $\mathrm{Set}^{\to}$ to the function $f_t$, which does indeed, by the naturality equation $f_t \circ (u_1, \ldots, u_n) = u \circ f_s$, satisfy that

$$x_1 \in u_1(X_1), \ldots, x_n \in u_n(X_n) \text{ implies } f_t(x_1, \ldots, x_n) \in u(X).$$

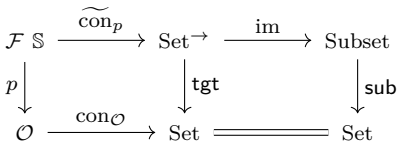
Observe that this morphism $\mathsf{tgt} \to \mathsf{sub}$ has a right adjoint




(A.17)

given by the functor interpreting a subset $U \subseteq X$ as a monic function $U \to X$.

**Corollary A.15.** *The composite morphism*




(A.18)

*generates an initial model of any bare gCFG $p$ in* $\mathsf{sub}$.

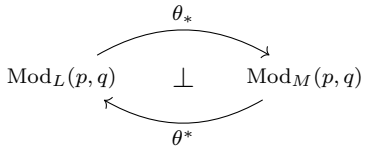
*Proof.* Since postcomposition with the morphism $\mathrm{im} : \mathsf{tgt} \to \mathsf{sub}$ induces a functor

$$\mathrm{Mod}(p, \mathrm{im}) : \mathrm{Mod}(p, \mathsf{tgt}) \to \mathrm{Mod}(p, \mathsf{sub})$$

which is a left adjoint, and as a left adjoint it preserves initial objects. □

In the case of a classical context-free grammar $G$ with associated functor $p : \mathcal{F}\,\mathbb{S} \to \mathcal{W}\,\Sigma$, the $\mathsf{sub}$-language is precisely the traditional language $\mathcal{L}_G^{\mathsf{sub}} = \mathcal{L}_G \subseteq \Sigma^*$ generated by the grammar, and more generally for a gCFG with start symbol $S \sqsubset A$ one recovers the language of constants $\mathcal{L}_G \subseteq \mathrm{con}\, A$ in the sense of Def. 1.15 as the $\mathsf{sub}$-language. Equations such as (A.2) and (A.3) are automatically satisfied as instances of the master equation (A.9), by virtue of the fact that the initial model in $\mathsf{sub} : \mathrm{Subset} \to \mathrm{Set}$ is exact (Prop. A.9) and the concrete description of the pushforward and fiberwise coproduct as image and union.

Both of these languages in $\mathsf{tgt}$ and $\mathsf{sub}$ can also be adapted relative to a fixed interpretation of the base operad $M : \mathcal{O} \to \mathrm{Set}$ by pushing forward the respective initial models along the unique natural transformation $!_M : \mathrm{con}_{\mathcal{O}} \Rightarrow M$. Abstractly, this may be seen as a consequence of the fact that both $\mathsf{tgt}$ and $\mathsf{sub}$ are bifibrations, so that any natural transformation $\theta : L \Rightarrow M$ induces an adjunction




(A.19)

between the respective categories of models (for $q = \mathsf{tgt}$ and $\mathsf{sub}$), and as a left adjoint the pushforward functor $\theta_*$ preserves initial objects. Concretely, in the case of $\mathsf{tgt}$, the language relative to $M$ interprets a gCFG $G$ with start symbol $S \sqsubset A$ as the composite arrow

$$\mathcal{L}_G^{M,\mathsf{tgt}} \quad := \quad \mathrm{con}_{\mathcal{F}\,\mathbb{S}}\, S \longrightarrow \mathrm{con}_{\mathcal{O}}\, A \longrightarrow M(A)$$

mapping every closed derivation $\alpha : S$ in $\mathcal{F}\,\mathbb{S}$ to the interpretation $M(c)$ of the constant $c = p(\alpha) : A$ in $\mathcal{O}$ that it derives.

For one more example of a $q$-language relative to a base interpretation, let us recall the notion of *Chomsky algebra* introduced by Grathwohl, Henglein, and Kozen [GHK17], defined as an idempotent semiring that is algebraically closed in the sense that every finite system of polynomial inequalities over the semiring has a least solution in the semiring. It is possible to

associate a polynomial refinement system to any idempotent semiring, as follows. Given an arbitrary (not necessarily idempotent) semiring $(R, +, \cdot, 0, 1)$, one can construct an operad $\mathcal{R}$ equipped with a functor $\pi_R : \mathcal{R} \to \mathcal{W}(R, \cdot, 1)$ into the operad of spliced arrows over its underlying multiplicative monoid seen as a one-object category. The operad $\mathcal{R}$ has elements of $R$ as colors, and operations $x_1, \ldots, x_n \to y$ given by spliced elements $a_0 - a_1 - \ldots - a_n$ such that $a_0 x_1 a_1 \ldots x_n a_n \leq y$, for the natural ordering defined by $x \leq y$ iff $\exists a. x + a = y$. The functor $\pi_R$ always has pushforwards, given by spliced multiplication. When $+$ is idempotent, $\pi_R$ will moreover have fiberwise coproducts, corresponding to joins with respect to $\leq$, and hence yield a polynomial refinement system.

Now, suppose given a gCFG $G = (\mathcal{O}, \mathbb{S}, S, p)$, a Chomsky algebra $(R, +, \cdot, 0, 1)$, and a functor $M : \mathcal{O} \to \mathcal{W}(R, \cdot, 1)$. (Most commonly, i.e., for a classical CFG, $\mathcal{O}$ will be $\mathcal{F}\, \mathbb{B}_\Sigma$ and $M$ will be derived from a function $\Sigma \to R$ assigning every letter of the alphabet an element of the semiring.) Any model of $G$ in $\pi_R$ with base interpretation $M$ is a solution to a finite system of polynomial inequalities determined by the production rules of $G$ and by the functor $M$. Hence there is an initial model by the defining assumption on a Chomsky algebra, i.e., $\mathcal{L}_G^{M,\pi_R}$ exists.

To conclude this preliminary exploration of the concept of $q$-language, we will now explain how some mild conditions on $q$ suffice to guarantee the existence of initial models, as well as to establish abstract versions of both the weak and strong translation principles. Although we are focused on modelling gCFGs, much of the story makes sense at a greater level of generality. Thus suppose given three functors of operads $p_1 : \mathcal{D}_1 \to \mathcal{O}$, $p_2 : \mathcal{D}_2 \to \mathcal{O}$ and $T : \mathcal{D}_1 \to \mathcal{D}_2$ defining a commutative diagram

$$\begin{array}{ccc} \mathcal{D}_1 & \xrightarrow{\quad T \quad} & \mathcal{D}_2 \\ & {\scriptstyle p_1} \searrow \quad \swarrow {\scriptstyle p_2} & \\ & \mathcal{O} & \end{array} \tag{A.20}$$

in the category $\mathsf{Oper}$, which we summarize by writing $T : p_1 \to p_2$. For any fixed functor of operads $M : \mathcal{O} \to \mathcal{B}$, precomposition with $T$ induces a functor of categories

$$T^* \quad : \quad \mathrm{Mod}_M(p_2, q) \longrightarrow \mathrm{Mod}_M(p_1, q)$$

which transports every model $(M, \tilde{M}) : p_2 \to q$ to the model $(M, \tilde{M} \circ T) : p_1 \to q$. Now, our aim is to show that under certain conditions on $q$, the functor $T^*$ has a left adjoint $T_*$. Hence, we will be in a situation where every functor $T : p_1 \to p_2$ as above induces an adjunction

$$\mathrm{Mod}_M(p_1, q) \underset{T^*}{\overset{T_*}{\rightleftarrows}} \mathrm{Mod}_M(p_2, q) \qquad \bot \tag{A.21}$$

between the corresponding categories of $q$-models relative to $M$, from which we will be able to extract various important consequences.

A sufficient condition for the existence of this left adjoint is that $q$ has pushforwards and has all (small) *fiberwise colimits,* in the natural sense generalizing fiberwise coproducts. Formally, we say that a functor $q : \mathcal{E} \to \mathcal{B}$ has **fiberwise colimits** when any diagram $D : J \to \mathcal{E}_X$ of unary operations lying in the fiber over some object $X \in \mathcal{B}$ has a colimit $(\underline{\mathsf{colim}}\, D, (in_i : D(i) \to \underline{\mathsf{colim}}\, D)_{i \in D})$ in the fiber category $\mathcal{E}_X$, and moreover the universal

property of this colimiting cocone extends to the total operad in the sense that precomposition with the $in_i$ induces a bijection

$$\mathcal{E}_f(\Theta, \underline{\mathsf{colim}}\, D, \Theta'; S) \cong \lim \mathcal{E}_f(\Theta, D(-), \Theta'; S)$$

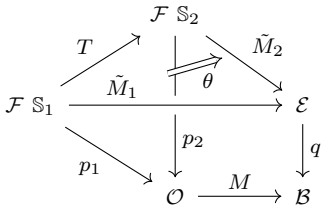
for every compatible operation $f$ of $\mathcal{B}$. (Note that if $q$ is an opfibration, then it has fiberwise colimits just in case every fiber category $\mathcal{E}_X$ is cocomplete and all of the pushforward functors $f_* : \mathcal{E}_{X_1} \times \cdots \times \mathcal{E}_{X_n} \to \mathcal{E}_X$ are cocontinuous.)

**Proposition A.16.** *Assume that $q : \mathcal{E} \to \mathcal{B}$ has pushforwards and fiberwise colimits. Let $p_1$, $p_2$, $T$, and $M$ be as above. Then the pullback functor $T^* : \mathrm{Mod}_M(p_2, q) \to \mathrm{Mod}_M(p_1, q)$ has a left adjoint $T_* : \mathrm{Mod}_M(p_1, q) \to \mathrm{Mod}_M(p_2, q)$.*

Before proving the proposition by showing how to construct the left adjoint, let us describe a few of its consequences.

*Existence of q-languages for all gCFGs.* Suppose given any gCFG $G = (\mathcal{O}, \mathbb{S}, S, p)$ and a base interpretation $M : \mathcal{O} \to \mathcal{B}$. Start by constructing an initial model $\tilde{M}$ of the underlying "discrete" grammar $|G| = (\mathcal{O}, |\mathbb{S}|, S, p \circ incl)$, where $incl : \mathcal{F}\ |\mathbb{S}| \to \mathcal{F}\ \mathbb{S}$ is the inclusion functor we described earlier. (Viewed as a grammar, $|G|$ has the same non-terminals and start symbol as $G$, but no production rules.) To construct the initial model of $|G|$ relative to $M$, it suffices to take $\tilde{M}$ to map every non-terminal $R \sqsubset^p A$ to the initial object $0_{M\,A} \in \mathcal{E}_{M\,A}$ of the appropriate fiber. Then we push $\tilde{M}$ forward along $incl$. Since left adjoints preserve initial objects, $incl_*\, \tilde{M}$ will be an initial model of $p$, and we have $\mathcal{L}_G^{M,q} := incl_*\, \tilde{M}\, S$.

*Translation principles.* Suppose given a pair of gCFGs $G_1 = (\mathcal{O}, \mathbb{S}_1, S_1, p_1)$ and $G_2 = (\mathcal{O}, \mathbb{S}_2, S_2, p_2)$ over the same operad, together with a translation $T : G_1 \to G_2$ in the sense of Definition 1.10. Fix a base interpretation $M : \mathcal{O} \to \mathcal{B}$. By the preceding, we know that there exist functors $\tilde{M}_1 : \mathcal{F}\ \mathbb{S}_1 \to \mathcal{E}$ and $\tilde{M}_2 : \mathcal{F}\ \mathbb{S}_2 \to \mathcal{E}$ such that $\tilde{M}_1$ and $\tilde{M}_2$ are initial objects in $\mathrm{Mod}_M(p_1, q)$ and $\mathrm{Mod}_M(p_2, q)$ respectively. Initiality of $\tilde{M}_1$ implies that there exists a unique morphism of models $\theta : \tilde{M}_1 \Rightarrow T^*\, \tilde{M}_2$, as depicted below:



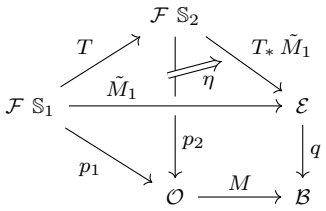

(A.22)

Instantiating this natural isomorphism at $S_1$ and applying the assumption that $T$ preserves the start symbol, we immediately obtain a morphism

$$\mathcal{L}_{G_1}^{M,q} \equiv \tilde{M}_1(S_1) \xrightarrow{\theta_{S_1}} \tilde{M}_2(T\, S_1) = \tilde{M}_2(S_2) \equiv \mathcal{L}_{G_2}^{M,q} \tag{A.23}$$

in the fiber category $\mathcal{E}_{M\,A}$, where $p_1(S_1) = p_2(S_2) = A$. This may be seen as a categorification of the weak translation principle, reducing to $\mathcal{L}_{G_1} \subseteq \mathcal{L}_{G_2}$ in the case that $q = \mathsf{sub}$ and $M = \mathrm{con}_{\mathcal{O}}$. Moreover, we know (again since pushforward preserves initial objects) that $\tilde{M}_2$ is isomorphic to the pushforward of $\tilde{M}_1$ along $T : p_1 \to p_2$, and that $\theta$ is equal to the composition of this isomorphism $T_*\, \tilde{M}_1 \cong \tilde{M}_2$ with the unit $\eta : \tilde{M}_1 \Rightarrow T^*\, T_*\, \tilde{M}_1$ of the

adjunction (A.21), as depicted below:




(A.24)

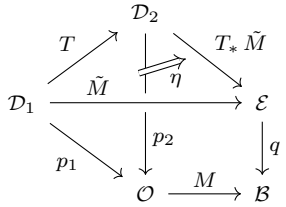
The last piece of the puzzle will be to show that when the functor $T$ is fully faithful then $\eta$ is invertible, although this relies on an explicit formula for the pushforward functor $T_*\,\tilde{M}_1$. From invertibility of $\eta$, we derive an isomorphism of $q$-languages

$$\mathcal{L}^{M,q}_{G_1} \equiv \tilde{M}_1(S_1) \overset{\eta_{S_1}}{\underset{\sim}{\Longrightarrow}} T^*\,T_*\,\tilde{M}_1(S_1) \cong \tilde{M}_2(S_2) \equiv \mathcal{L}^{M,q}_{G_2} \tag{A.25}$$

which may be seen as a categorification of the strong translation principle.[8]

Diagram (A.24) is suggestive of a Kan extension, and indeed the computation that we describe below of the pushforward model $T_*\,\tilde{M}_1$ may be seen as a kind of fibered and operadic generalization of the usual construction of the left Kan extension $\mathrm{Lan}_F\,G : \mathcal{C}_2 \to \mathcal{E}$ of a functor of categories $G : \mathcal{C}_1 \to \mathcal{E}$ along another functor of categories $F : \mathcal{C}_1 \to \mathcal{C}_2$, as a pointwise colimit in $\mathcal{E}$. The strong translation principle may then be seen as a generalization of the fact that if $F$ is fully faithful, then the unit 2-cell $\eta : G \Rightarrow \mathrm{Lan}_F\,G \circ F$ of such a pointwise left Kan extension is invertible [Mac98, Corollary X.3.3, p.239].

*Proof of Proposition A.16.* We suppose given an arbitrary commutative triangle of functors of operads $T : p_1 \to p_2$ as in (A.20) above, together with a target functor $q : \mathcal{E} \to \mathcal{B}$ that we assume to have pushforwards and fiberwise colimits, and a fixed base interpretation functor $M : \mathcal{O} \to \mathcal{B}$. We also suppose given an arbitrary (not necessarily initial) model of $p_1$ in $q$ relative to $M$, corresponding to a functor $\tilde{M} : \mathcal{D}_1 \to \mathcal{E}$ such that $q \circ \tilde{M} = M \circ p_1$. Our aim is to construct a functor $T_*\,\tilde{M} : \mathcal{D}_2 \to \mathcal{E}$ defining a model of $p_2$ in $q$ relative to $M$, equipped with a universal morphism of models $\eta : \tilde{M} \Rightarrow T^*\,T_*\,\tilde{M}$, as depicted below:




(A.26)

To define $T_*\,\tilde{M}$, we map each color $R \sqsubset^{p_2} A$ of $\mathcal{D}_2$ to the fiberwise colimit

$$T_*\,\tilde{M}\,R \quad := \quad \underline{\mathsf{colim}}\,D^{\tilde{M}}_R \tag{A.27}$$

of a certain diagram

$$D^{\tilde{M}}_R \quad : \quad T/R \longrightarrow \mathcal{E}_{M\,A}$$

defined as follows. The (relative) "operadic slice" category $T/R$ has objects given by tuples

$$(R_1, \ldots, R_n; \alpha)$$

[8]Indeed, in Proposition 1.11 we already tried to state the strong translation principle in a proof-relevant way, by mentioning that the grammars generate "isomorphic sets of parse trees for the arrows in the language". Formally, this is really an instance of the strong translation principle for $q$-languages, with $q = \mathsf{tgt}!$

consisting of a sequence of colors $R_1, \dots, R_n$ of $\mathcal{D}_1$ and an operation $\alpha : TR_1, \dots, TR_n \to R$ of $\mathcal{D}_2$, and arrows

$$(\gamma_1, \dots, \gamma_k) : (R_1, \dots, R_n; \alpha) \longrightarrow (S_1, \dots, S_k; \beta)$$

given by a sequence of operations $\gamma_1 : \Gamma_1 \to S_1, \dots, \gamma_k : \Gamma_k \to S_k$ of $\mathcal{D}_1$ such that $\alpha = \beta \circ (T\gamma_1, \dots, T\gamma_k)$ (which implies that $\Gamma_1, \dots, \Gamma_k$ is a composition of $R_1, \dots, R_n$ into $k$ parts). The functor $D_R^{\tilde{M}}$ is defined on objects by the pushforward formula

$$D_R^{\tilde{M}}(R_1, \dots, R_n; \alpha) \quad := \quad Mg(\tilde{M}R_1, \dots, \tilde{M}R_n) \tag{A.28}$$

where $g = p_2(\alpha)$ is the underlying image of $\alpha$ in $\mathcal{O}$. (This pushforward is well-defined and gives an object in the fiber $\mathcal{E}_{M\,A}$, by the assumption that $\alpha : TR_1, \dots, TR_n \to R$ and that $p_1 = p_2 \circ T$ and $q \circ \tilde{M} = M \circ p_1$.) It extends to the arrows of $T/R$ by applying the action of $\tilde{M}$ on operations of $\mathcal{D}_1$, as well as the laws of pushforwards. Finally, $T_*\,\tilde{M}$ extends to a functor $\mathcal{D}_2 \to \mathcal{E}$ by the functoriality of the slicing construction $T/-$ and the fact that pushforwards preserve fiberwise colimits.

Now, the natural transformation $\eta$ is given by taking

$$\eta_R \quad := \quad \tilde{M}R \cong Mid_A(\tilde{M}R) \longrightarrow \underline{\mathsf{colim}}\, D_R^{\tilde{M}} \tag{A.29}$$

to be the canonical injection into the colimit associated to the object $(R, id_{TR})$ of $T/R$, composed with a canonical isomorphism of the form (A.7) (since $Mid_A = id_{MA}$). Universality says that any model $\tilde{L} : p_2 \to q$ over $M$ equipped with a morphism of models $\theta : \tilde{M} \Rightarrow T^*\,\tilde{L}$ factors uniquely via $\eta$ and a morphism of models $\theta' : T_*\,\tilde{M} \Rightarrow \tilde{L}$. The latter corresponds to a natural family of arrows

$$\theta'_R \quad : \quad \underline{\mathsf{colim}}\, D_R^{\tilde{M}} \Longrightarrow \tilde{L}\,R$$

in $\mathcal{E}_{M\,A}$ for every color $R \sqsubset^{p_2} A$ of $\mathcal{D}_2$, the definition of which is entirely forced by the universal property of the colimit, the definition of the diagram $D_R^{\tilde{M}}$, and the input morphism $\theta : \tilde{M} \Rightarrow T^*\,L$. □

We now make two remarks by inspection of the formula for the pushforward model $T_*\,\tilde{M}$. First, when the functor $T : \mathcal{D}_1 \to \mathcal{D}_2$ is fully faithful, the relative slice category $T/TR$ is isomorphic to the slice category $\mathcal{D}_1/R$ for every color $R$ of $\mathcal{D}_1$. In particular, $(R, id_{TR})$ is a terminal object of $T/TR$ when $T$ is fully faithful. From this follows that the unit $\eta_R$ defined by (A.29) is invertible, and as a consequence we get the strong translation principle (A.25).

Second, although we assumed that $q$ has all fiberwise colimits, clearly the cardinalities of the indexing categories $T/R$ of the colimit diagrams are bounded to an extent by the cardinalities of $\mathcal{D}_1$ and $\mathcal{D}_2$. In particular, it suffices for $q$ to have *countable* fiberwise colimits to be able to define the pushforward

$$T_* \quad : \quad \mathrm{Mod}_M(p_1, q) \longrightarrow \mathrm{Mod}_M(p_2, q)$$

induced by a translation $T : G_1 \to G_2$ between any pair of gCFGs, since the respective operads $\mathcal{D}_1 = \mathcal{F}\,\mathbb{S}_1$ and $\mathcal{D}_2 = \mathcal{F}\,\mathbb{S}_2$ are freely generated from finite species.

We summarize the situation below.

**Corollary A.17.** *Let $q : \mathcal{E} \to \mathcal{B}$ be a functor with pushforwards and countable fiberwise colimits. Then every gCFG $G$ over an operad $\mathcal{O}$ generates a $q$-language $\mathcal{L}_G^{M,q}$ relative to any base interpretation $M : \mathcal{O} \to \mathcal{B}$. Moreover, $q$-languages enjoy both the weak and the strong translation principles, as expressed by the morphism* (A.23) *and the isomorphism* (A.25).

Of course it remains to establish the expected closure properties for $q$-gCFLs (cf. Proposition 1.19), and to show that the class of $q$-gCFLs interacts as expected with an appropriately formulated class of $q$-regular languages (cf. Corollary 3.7), which we leave as (unsolved!) exercises for the reader. We hope to have demonstrated that there is some interest in this slightly more abstract way of thinking about the languages generated by context-free grammars, complementary to the more concrete approach taken in the main body of the paper.

## Contents